М. Э. Абрамян

Programming Taskbook

Электронный задачник по программированию

Версия 4.6

Ростов-на-Дону
2007

Дата генерации PDF-документа: 23.02.2007.

Общее описание

Электронный задачник Programming Taskbook предназначен для обучения программированию на языках Pascal, Visual Basic, C++, C#, Visual Basic .NET. Он содержит 1000 учебных заданий, охватывающих все основные разделы базового курса программирования: от скалярных типов и управляющих операторов до сложных структур данных и рекурсивных алгоритмов.

Версия 4.6 задачника Programming Taskbook реализована для следующих сред: Borland Delphi 3.0–7.0 и 2006, в частности, Turbo Delphi 2006 for Windows, Microsoft Visual Basic 5.0 и 6.0, Borland C++Builder 4.0 и 5.0, Microsoft Visual C++ 6.0, Microsoft Visual Studio .NET 2003 и 2005 (языки Visual C++, Visual Basic .NET и Visual C# .NET). Кроме того, задачник включен в учебную систему программирования Pascal ABC (автор доц. С. С. Михалкович), образуя единый программный комплекс Pascal ABC & Programming Taskbook.

Задачник Programming Taskbook предоставляет учащимся следующие возможности:

- отображение на экране текста задания и связанных с ним данных;
- демонстрация правильных результатов для каждого задания;
- предоставление исходных данных программе учащегося;
- дополнительный контроль за операциями ввода-вывода;
- проверка правильности результатов, полученных программой;
- запись в особый *файл результатов* информации о каждом тестовом испытании программы;
- регистрация задания как выполненного после проведения серии успешных тестовых испытаний программы.

Использование электронного задачника существенно ускоряет процесс выполнения заданий, так как избавляет учащегося от дополнительных усилий по организации ввода-вывода, что особенно удобно при обработке массивов, строк, файлов и динамических структур. Предлагая учащемуся готовые исходные данные, задачник акцентирует его внимание на разработке и программной реализации *алгоритма* решения заданий, причем разнообразие исходных данных обеспечивает надежное *тестирование* предложенного алгоритма.

Задачник содержит следующие группы учебных заданий (в скобках указано количество заданий для каждой группы):

- Begin — ввод и вывод данных, оператор присваивания (40),

- Integer — целые числа (30),
- Boolean — логические выражения (40),
- If — условный оператор (30),
- Case — оператор выбора (20),
- For — цикл с параметром (40),
- While — цикл с условием (30),
- Series — последовательности (40),
- Proc — процедуры и функции (60),
- Minmax — минимумы и максимумы (30),
- Array — одномерные массивы (140),
- Matrix — двумерные массивы (100),
- String — символы и строки (70),
- File — двоичные файлы (90),
- Text — текстовые файлы (60),
- Param — составные типы данных в процедурах и функциях (70),
- Recur — рекурсия (30),
- Dynamic — динамические структуры данных (80).

PDF-версия задачника Programming Taskbook содержит формулировки всех учебных заданий. Формулировки отформатированы в соответствии с печатным вариантом задачника, который приведен в методических указаниях [1–3] и книгах [4–6].

Задания, помеченные символом «°», можно выполнять в свободно распространяемом *мини-варианте* задачника PT4Mini–250. Доступными для выполнения в мини-варианте являются 250 заданий, в том числе все задания групп Begin, Integer, Boolean, а также 140 избранных заданий из других групп задачника. Следует отметить, что в мини-варианте задачника можно выполнять все задания, решения которых даются в книгах «Основы программирования на языке Паскаль», «Практикум по программированию на языке Паскаль», и «Практикум по программированию на языках C# и VB .NET». Ниже приводится список всех заданий, доступных для выполнения в мини-варианте PT4Mini–250:

Begin1–Begin40, Integer1–Integer30, Boolean1–Boolean40, If4, If6, If8, If12, If22, If26, Case2, Case4, Case9–Case10, Case18, For5, For12–For13, For15–For16, For19–For20, For33, For36, While1–While2, While4, While7, While11–While12, While22–While23, Series1, Series15–Series17, Series19, Series21, Series30, Proc4, Proc8, Proc10, Proc20–Proc21, Proc25, Proc40, Minmax1, Minmax4, Minmax6,

Minmax12, Minmax19, Minmax22, Array4, Array7, Array16, Array32, Array47, Array54, Array63, Array71, Array79, Array89, Array92, Array108, Array112, Array116, Array134, Matrix7, Matrix24, Matrix36, Matrix53, Matrix74, Matrix82, Matrix88, Matrix100, String9–String10, String19, String29, String41, String44, String63, String70, File2, File10, File25, File27, File41, File43, File48, File50, File58, File61, File63, File67, File74, Text1, Text4, Text16, Text21, Text24, Text34, Text38, Text42, Text44, Text57, Param1, Param17, Param30, Param40, Param49, Param53, Param59–Param61, Recur1, Recur4–Recur5, Recur10, Recur14–Recur18, Recur21, Recur25, Recur27, Dynamic2–Dynamic3, Dynamic5, Dynamic8–Dynamic12, Dynamic25, Dynamic30, Dynamic49, Dynamic55, Dynamic59, Dynamic63, Dynamic70, Dynamic74, Dynamic78.

В состав задачника входят следующие вспомогательные программные модули:

- PT4Demo — позволяет просмотреть в демонстрационном режиме все задания, включенные в задачник;
- PT4Load — обеспечивает генерацию программы-шаблона для требуемого учебного задания и ее немедленную загрузку в выбранную среду программирования;
- PT4Result — предназначен для расшифровки, анализа и отображения на экране содержимого *файла результатов*, в который заносятся сведения о ходе выполнения заданий

Эти модули реализованы в виде отдельных программ-утилит, доступных из меню задачника «Пуск | Программы | Programming Taskbook 4». Кроме того, эти модули можно вызывать из меню тех программных сред, в которых используется задачник (соответствующие команды находятся в подменю «Tools» или «Add-Ins»).

Дополнением к задачнику Programming Taskbook является свободно распространяемый комплекс «Teacher Pack for Programming Taskbook 4». Он содержит компоненты электронного задачника, которые предназначены для преподавателя программирования и призваны упростить подготовку и проведение групповых практических занятий. Комплекс состоит из следующих элементов:

- PTVarMaker.exe — программа «Конструктор вариантов». Позволяет автоматически генерировать *индивидуальный набор заданий* для каждого учащегося, а также создавать *контрольные файлы*, обеспечивающие регистрацию задания как выполненного только после проверки преподавателем текста программы, решающей это задание;

- PT4Teach.exe — программа «Контрольный центр преподавателя». Предназначена для подготовки каталогов учащихся к проведению занятий, обеспечения дополнительного контроля за процессом выполнения заданий, получения и анализа информации о результатах выполнения заданий, резервного копирования важнейших файлов из каталогов учащихся;
- TeacherPack.chm — справочная система «Teacher Pack Info». СОдержит полное описание программ PTVarMaker и PT4Teach, сведения о вспомогательных файлах, используемых задачником Programming Taskbook, и рекомендации по проведению групповых занятий.

Имеется также вариант данного комплекса «Teacher Pack for Pascal ABC», ориентированный на совместное использование с учебной системой программирования Pascal ABC (версии не ниже 2.5) и обладающий теми же возможностями.

Замечания о формулировках заданий

В формулировках заданий не используются понятия и имена, специфические для конкретного языка программирования.

Если о типе исходных или результирующих числовых данных в задании ничего не сказано, то предполагаются *вещественные* данные. Исключение составляет группа заданий Dynamic, в которой все числовые данные считаются *целыми*, и в формулировках заданий это особо не оговаривается.

При обработке наборов *вещественных* чисел следует предполагать, что все элементы набора являются *различными* (таким образом, любой набор вещественных чисел содержит единственный минимальный и единственный максимальный элемент). В наборах *целых* чисел могут присутствовать *одинаковые* элементы; в частности, наборы целых чисел могут содержать несколько минимальных и максимальных элементов. Аналогичные предположения справедливы для числовых массивов, а также для файлов, содержащих числовые данные.

Если в задании не указан максимальный размер исходных массивов, то его можно считать равным 10 для одномерных и 10×10 для двумерных массивов.

При описании элементов одномерных и двумерных массивов используется понятие *порядкового номера элемента*, причем начальный элемент массива A размера N всегда имеет порядковый номер 1 и обозначается в формули-

ровках заданий как A_1, а конечный элемент этого же массива имеет порядковый номер N и обозначается как A_N. Аналогично, начальный элемент двумерного массива B обозначается как $B_{1,1}$. Кроме того, понятие порядкового номера применяется к *строкам* и *столбцам* двумерных массивов (матриц): начальная строка и начальный столбец матрицы размера $M \times N$ имеют порядковый номер 1, конечная строка — номер M, а конечный столбец — номер N. Подобный подход не зависит от выбора языка программирования и соответствует традиционно используемой в математике нумерации элементов векторов и матриц.

Максимальный размер исходных *файлов* не указывается, поэтому при решении заданий на файлы не следует использовать вспомогательные массивы, содержащие все элементы исходных файлов, однако допускается использование *вспомогательных файлов*. Все исходные файлы считаются существующими, за исключением специально оговоренных случаев, в которых существование исходных файлов требуется проверять в ходе выполнения задания.

Под *размером* двоичного типизированного файла всегда подразумевается количество содержащихся в нем *элементов* указанного типа (а не количество байтов, как это принято в операционной системе). Как и для элементов массива, для элементов двоичного файла в формулировках заданий применяется «естественная» нумерация: первый элемент файла имеет порядковый номер 1.

Если в используемом языке программирования отсутствует понятие «процедура», то под процедурой в формулировках заданий групп Proc, Param и Dynamic надо понимать *функцию, не имеющую возвращаемого значения*.

Формулировки заданий группы Dynamic приводятся в двух вариантах: первый вариант (см. раздел «Динамические структуры данных») ориентирован на применение *указателей* и предназначен для языков Паскаль и C++, второй вариант (см. раздел «Динамические структуры данных (.NET)») ориентирован на использование *объектной модели платформы .NET* и предназначен для языков C# и Visual Basic .NET. Заметим,что в варианте задачника для языка Visual Basic группа Dynamic отсутствует.

Ввод исходных данных и вывод результатов

При выполнении учебных заданий с использованием электронного задачника Programming Taskbook ввод-вывод данных должен осуществляться с помощью *специальных процедур или функций*, реализованных в самом задачнике и доступных для программы, выполняющей задание. Это связано с нем,

что исходные данные, предоставляемые программе учащегося, генерируются самим задачником, а результаты, полученные программой, должны быть проверены задачником на правильность.

Для получения краткой справки о средствах ввода-вывода, которые можно использовать для выбранного языка программирования, достаточно, находясь в окне задачника, нажать клавишу [F1] или кнопку со знаком вопроса «?» в правой части заголовка окна (окно задачника можно отобразить на экране, используя модуль PT4Demo или запустив на выполнение проект-заготовку, созданный с помощью модуля PT4Load).

Ниже приводится описание процедур/функций ввода-вывода для каждого из языков программирования, доступных в задачнике Programming Taskbook версии 4.6. Более подробную информацию о вводе-выводе данных (в частности, об особенностях ввода-вывода для файлов и динамических структур) можно получить с помощью гипертекстовой справочной системы PT4Info, входящей в состав версии 4.6 задачника Programming Taskbook.

Язык Паскаль

Ввод исходных данных:

```
procedure GetB(var A: boolean);
procedure GetN(var A: integer);
procedure GetR(var A: real);
procedure GetC(var A: char);
procedure GetS(var A: string);
procedure GetP(var A: PNode);
```

Вывод результатов:

```
procedure PutB(A: boolean);
procedure PutN(A: integer);
procedure PutR(A: real);
procedure PutC(A: char);
procedure PutS(A: string);
procedure PutP(A: PNode);
```

По поводу типа PNode см. описание группы заданий Dymanic.

При использовании задачника в среде программирования Pascal ABC для ввода-вывода данных можно использовать стандартные процедуры Read и Write.

Язык C++

Ввод исходных данных:

```
void GetB(bool& a);
void GetN(int& a);
void GetD(double& a);
void GetC(char& a);
void GetS(char* a);
void GetS(string& a);
void GetP(TNode*& a);
```

Вывод результатов:

```
void PutB(bool a);
void PutN(int a);
void PutD(double a);
void PutC(char a);
void PutS(char* a);
void PutS(string a);
void PutP(TNode* a);
```

Для ввода-вывода данных может также использоваться *поток ввода-вывода* pt, позволяющий добиться более компактной записи. Например, вместо последовательности вызовов функций GetN(a); GetD(b); GetS(s); достаточно указать один оператор чтения из потока: pt >> a >> b >> s;

По поводу типа PNode см. описание группы заданий Dymanic.

Язык Visual Basic

Ввод исходных данных:

```
Public Sub GetB(ByRef A As Boolean)
Public Sub GetN(ByRef A As Integer)
Public Sub GetD(ByRef A As Double)
Public Sub GetS(ByRef A As String)
```

Вывод результатов:

```
Public Sub PutB(ByVal A As Boolean)
Public Sub PutN(ByVal A As Integer)
Public Sub PutD(ByVal A As Double)
Public Sub PutS(ByVal A As String)
```

Процедуры GetS и PutS должны использоваться как для ввода-вывода строк, так и для ввода-вывода символов (поскольку в языке Visual Basic отсутствует особый символьный тип данных).

Языки платформы .NET (C# и VB.NET)

Ввод исходных данных (C#):

```
public static bool GetBool();
public static int GetInt();
public static double GetDouble();
public static char GetChar();
public static string GetString();
public static Node GetNode();
```

Ввод исходных данных (VB.NET):

```
Public Shared Function GetBool() As Boolean
Public Shared Function GetInt() As Integer
Public Shared Function GetDouble() As Double
Public Shared Function GetChar() As Char
Public Shared Function GetString() As String
Public Shared Function GetNode() As Node
```

Вывод результатов (C#):

```
public static void Put(params object[] a);
```

Вывод результатов (VB.NET):

```
Public Shared Sub Put(ParamArray a() As Object)
```

Операции ввода-вывода для языков платформы .NET реализованы в виде *классовых методов* специального класса PT, доступного для любой программы, выполняющей учебное задание. В указании имени класса PT при вызове этих методов нет необходимости, так как решение задания оформляется в виде метода Solve класса, который является *потомком* класса PT (и, следовательно, получает все методы ввода-вывода «по наследству»).

Метод Put является универсальным методом вывода. В нем можно указывать любое число выводимых параметров любого допустимого типа (а именно, любого из типов, для которого предусмотрен метод ввода).

По поводу типа Node см. описание группы заданий Dymanic для платформы .NET.

Ввод и вывод данных, оператор присваивания

Все входные и выходные данные в заданиях этой группы являются вещественными числами.

Begin1°. Дана сторона квадрата a. Найти его периметр $P = 4 \cdot a$.

Begin2°. Дана сторона квадрата a. Найти его площадь $S = a^2$.

Begin3°. Даны стороны прямоугольника a и b. Найти его площадь $S = a \cdot b$ и периметр $P = 2 \cdot (a + b)$.

Begin4°. Дан диаметр окружности d. Найти ее длину $L = \pi \cdot d$. В качестве значения π использовать 3.14.

Begin5°. Дана длина ребра куба a. Найти объем куба $V = a^3$ и площадь его поверхности $S = 6 \cdot a^2$.

Begin6°. Даны длины ребер a, b, c прямоугольного параллелепипеда. Найти его объем $V = a \cdot b \cdot c$ и площадь поверхности $S = 2 \cdot (a \cdot b + b \cdot c + a \cdot c)$.

Begin7°. Найти длину окружности L и площадь круга S заданного радиуса R:

$$L = 2 \cdot \pi \cdot R, \qquad S = \pi \cdot R^2.$$

В качестве значения π использовать 3.14.

Begin8°. Даны два числа a и b. Найти их *среднее арифметическое*: $(a + b)/2$.

Begin9°. Даны два неотрицательных числа a и b. Найти их *среднее геометрическое*, то есть квадратный корень из их произведения: $\sqrt{a \cdot b}$.

Begin10°. Даны два ненулевых числа. Найти сумму, разность, произведение и частное их квадратов.

Begin11°. Даны два ненулевых числа. Найти сумму, разность, произведение и частное их модулей.

Begin12°. Даны катеты прямоугольного треугольника a и b. Найти его гипотенузу c и периметр P:

$$c = \sqrt{a^2 + b^2}, \qquad P = a + b + c.$$

Begin13°. Даны два круга с общим центром и радиусами R_1 и R_2 ($R_1 > R_2$). Найти площади этих кругов S_1 и S_2, а также площадь S_3 кольца, внешний радиус которого равен R_1, а внутренний радиус равен R_2:

$$S_1 = \pi \cdot (R_1)^2, \qquad S_2 = \pi \cdot (R_2)^2, \qquad S_3 = S_1 - S_2.$$

В качестве значения π использовать 3.14.

Begin14°. Дана длина L окружности. Найти ее радиус R и площадь S круга, ограниченного этой окружностью, учитывая, что $L = 2 \cdot \pi \cdot R$, $S = \pi \cdot R^2$. В качестве значения π использовать 3.14.

Begin15°. Дана площадь S круга. Найти его диаметр D и длину L окружности, ограничивающей этот круг, учитывая, что $L = 2 \cdot \pi \cdot R$, $S = \pi \cdot R^2$. В качестве значения π использовать 3.14.

Begin16°. Найти расстояние между двумя точками с заданными координатами x_1 и x_2 на числовой оси: $|x_2 - x_1|$.

Begin17°. Даны три точки A, B, C на числовой оси. Найти длины отрезков AC и BC и их сумму.

Begin18°. Даны три точки A, B, C на числовой оси. Точка C расположена между точками A и B. Найти произведение длин отрезков AC и BC.

Begin19°. Даны координаты двух противоположных вершин прямоугольника: (x_1, y_1), (x_2, y_2). Стороны прямоугольника параллельны осям координат. Найти периметр и площадь данного прямоугольника.

Begin20°. Найти расстояние между двумя точками с заданными координатами (x_1, y_1) и (x_2, y_2) на плоскости. Расстояние вычисляется по формуле

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

Begin21°. Даны координаты трех вершин треугольника: (x_1, y_1), (x_2, y_2), (x_3, y_3). Найти его периметр и площадь, используя формулу для расстояния между двумя точками на плоскости (см. задание Begin20). Для нахождения площади треугольника со сторонами a, b, c использовать *формулу Герона*:

$$S = \sqrt{p \cdot (p - a) \cdot (p - b) \cdot (p - c)},$$

где $p = (a + b + c)/2$ — *полупериметр*.

Begin22°. Поменять местами содержимое переменных A и B и вывести новые значения A и B.

Begin23°. Даны переменные A, B, C. Изменить их значения, переместив содержимое A в B, B — в C, C — в A, и вывести новые значения переменных A, B, C.

Begin24°. Даны переменные A, B, C. Изменить их значения, переместив содержимое A в C, C — в B, B — в A, и вывести новые значения переменных A, B, C.

Begin25°. Найти значение функции $y = 3x^6 - 6x^2 - 7$ при данном значении x.

Begin26°. Найти значение функции $y = 4(x-3)^6 - 7(x-3)^3 + 2$ при данном значении x.

Begin27°. Дано число A. Вычислить A^8, используя вспомогательную переменную и три операции умножения. Для этого последовательно находить A^2, A^4, A^8. Вывести все найденные степени числа A.

Begin28°. Дано число A. Вычислить A^{15}, используя две вспомогательные пере-

менные и пять операций умножения. Для этого последовательно находить A^2, A^3, A^5, A^{10}, A^{15}. Вывести все найденные степени числа A.

Begin29°. Дано значение угла α в градусах ($0 < \alpha < 360$). Определить значение этого же угла в радианах, учитывая, что $180^\circ = \pi$ радианов. В качестве значения π использовать 3.14.

Begin30°. Дано значение угла α в радианах ($0 < \alpha < 2\cdot\pi$). Определить значение этого же угла в градусах, учитывая, что $180^\circ = \pi$ радианов. В качестве значения π использовать 3.14.

Begin31°. Дано значение температуры T в градусах Фаренгейта. Определить значение этой же температуры в градусах Цельсия. Температура по Цельсию T_C и температура по Фаренгейту T_F связаны следующим соотношением:

$$T_C = (T_F - 32)\cdot 5/9.$$

Begin32°. Дано значение температуры T в градусах Цельсия. Определить значение этой же температуры в градусах Фаренгейта. Температура по Цельсию T_C и температура по Фаренгейту T_F связаны следующим соотношением:

$$T_C = (T_F - 32)\cdot 5/9.$$

Begin33°. Известно, что X кг конфет стоит A рублей. Определить, сколько стоит 1 кг и Y кг этих же конфет.

Begin34°. Известно, что X кг шоколадных конфет стоит A рублей, а Y кг ирисок стоит B рублей. Определить, сколько стоит 1 кг шоколадных конфет, 1 кг ирисок, а также во сколько раз шоколадные конфеты дороже ирисок.

Begin35°. Скорость лодки в стоячей воде V км/ч, скорость течения реки U км/ч ($U < V$). Время движения лодки по озеру T_1 ч, а по реке (против течения) — T_2 ч. Определить путь S, пройденный лодкой (путь = время · скорость). Учесть, что при движении против течения скорость лодки уменьшается на величину скорости течения.

Begin36°. Скорость первого автомобиля V_1 км/ч, второго — V_2 км/ч, расстояние между ними S км. Определить расстояние между ними через T часов, если автомобили удаляются друг от друга. Данное расстояние равно сумме начального расстояния и общего пути, проделанного автомобилями; общий путь = время · суммарная скорость.

Begin37°. Скорость первого автомобиля V_1 км/ч, второго — V_2 км/ч, расстояние между ними S км. Определить расстояние между ними через T часов, если автомобили первоначально движутся навстречу друг другу. Данное

расстояние равно модулю разности начального расстояния и общего пути, проделанного автомобилями; общий путь = время · суммарная скорость.

Begin38°. Решить линейное уравнение $A \cdot x + B = 0$, заданное своими коэффициентами A и B (коэффициент A не равен 0).

Begin39°. Найти корни *квадратного уравнения* $A \cdot x^2 + B \cdot x + C = 0$, заданного своими коэффициентами A, B, C (коэффициент A не равен 0), если известно, что дискриминант уравнения положителен. Вывести вначале меньший, а затем больший из найденных корней. Корни квадратного уравнения находятся по формуле

$$x_{1,2} = (-B \pm \sqrt{D})/(2 \cdot A),$$

где D — *дискриминант*, равный $B^2 - 4 \cdot A \cdot C$.

Begin40°. Найти решение *системы линейных уравнений* вида

$$A_1 \cdot x + B_1 \cdot y = C_1,$$
$$A_2 \cdot x + B_2 \cdot y = C_2,$$

заданной своими коэффициентами A_1, B_1, C_1, A_2, B_2, C_2, если известно, что данная система имеет единственное решение. Воспользоваться формулами

$$x = (C_1 \cdot B_2 - C_2 \cdot B_1)/D, \qquad y = (A_1 \cdot C_2 - A_2 \cdot C_1)/D,$$

где $D = A_1 \cdot B_2 - A_2 \cdot B_1$.

Целые числа

Все входные и выходные данные в заданиях этой группы являются целыми числами. Все числа, для которых указано количество цифр (двузначное число, трехзначное число и т. д.), считаются положительными.

Integer1°. Дано расстояние L в сантиметрах. Используя операцию деления нацело, найти количество полных метров в нем (1 метр = 100 см).

Integer2°. Дана масса M в килограммах. Используя операцию деления нацело, найти количество полных тонн в ней (1 тонна = 1000 кг).

Integer3°. Дан размер файла в байтах. Используя операцию деления нацело, найти количество полных килобайтов, которые занимает данный файл (1 килобайт = 1024 байта).

Integer4°. Даны целые положительные числа A и B ($A > B$). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Используя операцию деления нацело, найти количество отрезков B, размещенных на отрезке A.

Integer5°. Даны целые положительные числа A и B ($A > B$). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Используя операцию взятия остатка от деления нацело, найти длину незанятой части отрезка A.

Integer6°. Дано двузначное число. Вывести вначале его левую цифру (десятки), а затем — его правую цифру (единицы). Для нахождения десятков использовать операцию деления нацело, для нахождения единиц — операцию взятия остатка от деления.

Integer7°. Дано двузначное число. Найти сумму и произведение его цифр.

Integer8°. Дано двузначное число. Вывести число, полученное при перестановке цифр исходного числа.

Integer9°. Дано трехзначное число. Используя одну операцию деления нацело, вывести первую цифру данного числа (сотни).

Integer10°. Дано трехзначное число. Вывести вначале его последнюю цифру (единицы), а затем — его среднюю цифру (десятки).

Integer11°. Дано трехзначное число. Найти сумму и произведение его цифр.

Integer12°. Дано трехзначное число. Вывести число, полученное при прочтении исходного числа справа налево.

Integer13°. Дано трехзначное число. В нем зачеркнули первую слева цифру и приписали ее справа. Вывести полученное число.

Integer14°. Дано трехзначное число. В нем зачеркнули первую справа цифру и приписали ее слева. Вывести полученное число.

Integer15°. Дано трехзначное число. Вывести число, полученное при перестановке цифр сотен и десятков исходного числа (например, 123 перейдет в 213).

Integer16°. Дано трехзначное число. Вывести число, полученное при перестановке цифр десятков и единиц исходного числа (например, 123 перейдет в 132).

Integer17°. Дано целое число, большее 999. Используя одну операцию деления нацело и одну операцию взятия остатка от деления, найти цифру, соответствующую разряду сотен в записи этого числа.

Integer18°. Дано целое число, большее 999. Используя одну операцию деления нацело и одну операцию взятия остатка от деления, найти цифру, соответствующую разряду тысяч в записи этого числа.

Integer19°. С начала суток прошло N секунд (N — целое). Найти количество полных минут, прошедших с начала суток.

Integer20°. С начала суток прошло N секунд (N — целое). Найти количество полных часов, прошедших с начала суток.

Integer21°. С начала суток прошло N секунд (N — целое). Найти количество секунд, прошедших с начала последней минуты.

Integer22°. С начала суток прошло N секунд (N — целое). Найти количество секунд, прошедших с начала последнего часа.

Integer23°. С начала суток прошло N секунд (N — целое). Найти количество полных минут, прошедших с начала последнего часа.

Integer24°. Дни недели пронумерованы следующим образом: 0 — воскресенье, 1 — понедельник, 2 — вторник, . . . , 6 — суббота. Дано целое число K, лежащее в диапазоне 1–365. Определить номер дня недели для K-го дня года, если известно, что в этом году 1 января было понедельником.

Integer25°. Дни недели пронумерованы следующим образом: 0 — воскресенье, 1 — понедельник, 2 — вторник, . . . , 6 — суббота. Дано целое число K, лежащее в диапазоне 1–365. Определить номер дня недели для K-го дня года, если известно, что в этом году 1 января было четвергом.

Integer26°. Дни недели пронумерованы следующим образом: 1 — понедельник, 2 — вторник, . . . , 6 — суббота, 7 — воскресенье. Дано целое число K, лежащее в диапазоне 1–365. Определить номер дня недели для K-го дня года, если известно, что в этом году 1 января было вторником.

Integer27°. Дни недели пронумерованы следующим образом: 1 — понедельник, 2 — вторник, . . . , 6 — суббота, 7 — воскресенье. Дано целое число K, лежащее в диапазоне 1–365. Определить номер дня недели для K-го дня года, если известно, что в этом году 1 января было субботой.

Integer28°. Дни недели пронумерованы следующим образом: 1 — понедельник, 2 — вторник, . . . , 6 — суббота, 7 — воскресенье. Дано целое число K, лежащее в диапазоне 1–365, и целое число N, лежащее в диапазоне 1–7. Определить номер дня недели для K-го дня года, если известно, что в этом году 1 января было днем недели с номером N.

Integer29°. Даны целые положительные числа A, B, C. На прямоугольнике размера $A \times B$ размещено максимально возможное количество квадратов со стороной C (без наложений). Найти количество квадратов, размещенных на прямоугольнике, а также площадь незанятой части прямоугольника.

Integer30°. Дан номер некоторого года (целое положительное число). Определить соответствующий ему номер столетия, учитывая, что, к примеру, началом 20 столетия был 1901 год.

Логические выражения

Во всех заданиях данной группы требуется вывести логическое значение TRUE, если приведенное высказывание для предложенных исходных данных является истинным, и значение FALSE в противном случае. Все числа, для которых указано количество цифр (двузначное число, трехзначное число и т. д.), считаются целыми положительными.

Boolean1°. Дано целое число A. Проверить истинность высказывания: «Число A является положительным».

Boolean2°. Дано целое число A. Проверить истинность высказывания: «Число A является нечетным».

Boolean3°. Дано целое число A. Проверить истинность высказывания: «Число A является четным».

Boolean4°. Даны два целых числа: A, B. Проверить истинность высказывания: «Справедливы неравенства $A > 2$ и $B \leq 3$».

Boolean5°. Даны два целых числа: A, B. Проверить истинность высказывания: «Справедливы неравенства $A \geq 0$ или $B < -2$».

Boolean6°. Даны три целых числа: A, B, C. Проверить истинность высказывания: «Справедливо двойное неравенство $A < B < C$».

Boolean7°. Даны три целых числа: A, B, C. Проверить истинность высказывания: «Число B находится между числами A и C».

Boolean8°. Даны два целых числа: A, B. Проверить истинность высказывания: «Каждое из чисел A и B нечетное».

Boolean9°. Даны два целых числа: A, B. Проверить истинность высказывания: «Хотя бы одно из чисел A и B нечетное».

Boolean10°. Даны два целых числа: A, B. Проверить истинность высказывания: «Ровно одно из чисел A и B нечетное».

Boolean11°. Даны два целых числа: A, B. Проверить истинность высказывания: «Числа A и B имеют одинаковую четность».

Boolean12°. Даны три целых числа: A, B, C. Проверить истинность высказывания: «Каждое из чисел A, B, C положительное».

Boolean13°. Даны три целых числа: A, B, C. Проверить истинность высказывания: «Хотя бы одно из чисел A, B, C положительное».

Boolean14°. Даны три целых числа: A, B, C. Проверить истинность высказывания: «Ровно одно из чисел A, B, C положительное».

Boolean15°. Даны три целых числа: A, B, C. Проверить истинность высказывания: «Ровно два из чисел A, B, C являются положительными».

Boolean16°. Дано целое положительное число. Проверить истинность высказывания: «Данное число является четным двузначным».

Boolean17°. Дано целое положительное число. Проверить истинность высказывания: «Данное число является нечетным трехзначным».

Boolean18°. Проверить истинность высказывания: «Среди трех данных целых чисел есть хотя бы одна пара совпадающих».

Boolean19°. Проверить истинность высказывания: «Среди трех данных целых чисел есть хотя бы одна пара взаимно противоположных».

Boolean20°. Дано трехзначное число. Проверить истинность высказывания: «Все цифры данного числа различны».

Boolean21°. Дано трехзначное число. Проверить истинность высказывания: «Цифры данного числа образуют возрастающую последовательность».

Boolean22°. Дано трехзначное число. Проверить истинность высказывания: «Цифры данного числа образуют возрастающую или убывающую последовательность».

Boolean23°. Дано четырехзначное число. Проверить истинность высказывания: «Данное число читается одинаково слева направо и справа налево».

Boolean24°. Даны числа A, B, C (число A не равно 0). Рассмотрев *дискриминант* $D = B^2 - 4 \cdot A \cdot C$, проверить истинность высказывания: «Квадратное уравнение $A \cdot x^2 + B \cdot x + C = 0$ имеет вещественные корни».

Boolean25°. Даны числа x, y. Проверить истинность высказывания: «Точка с координатами (x, y) лежит во второй координатной четверти».

Boolean26°. Даны числа x, y. Проверить истинность высказывания: «Точка с координатами (x, y) лежит в четвертой координатной четверти».

Boolean27°. Даны числа x, y. Проверить истинность высказывания: «Точка с координатами (x, y) лежит во второй или третьей координатной четверти».

Boolean28°. Даны числа x, y. Проверить истинность высказывания: «Точка с координатами (x, y) лежит в первой или третьей координатной четверти».

Boolean29°. Даны числа x, y, x_1, y_1, x_2, y_2. Проверить истинность высказывания: «Точка с координатами (x, y) лежит внутри прямоугольника, левая верхняя вершина которого имеет координаты (x_1, y_1), правая нижняя — (x_2, y_2), а стороны параллельны координатным осям».

Boolean30°. Даны целые числа a, b, c, являющиеся сторонами некоторого треугольника. Проверить истинность высказывания: «Треугольник со сторо-

нами a, b, c является равносторонним».

Boolean31°. Даны целые числа a, b, c, являющиеся сторонами некоторого треугольника. Проверить истинность высказывания: «Треугольник со сторонами a, b, c является равнобедренным».

Boolean32°. Даны целые числа a, b, c, являющиеся сторонами некоторого треугольника. Проверить истинность высказывания: «Треугольник со сторонами a, b, c является прямоугольным».

Boolean33°. Даны целые числа a, b, c. Проверить истинность высказывания: «Существует треугольник со сторонами a, b, c».

Boolean34°. Даны координаты поля шахматной доски x, y (целые числа, лежащие в диапазоне 1–8). Учитывая, что левое нижнее поле доски $(1, 1)$ является черным, проверить истинность высказывания: «Данное поле является белым».

Boolean35°. Даны координаты двух различных полей шахматной доски x_1, y_1, x_2, y_2 (целые числа, лежащие в диапазоне 1–8). Проверить истинность высказывания: «Данные поля имеют одинаковый цвет».

Boolean36°. Даны координаты двух различных полей шахматной доски x_1, y_1, x_2, y_2 (целые числа, лежащие в диапазоне 1–8). Проверить истинность высказывания: «Ладья за один ход может перейти с одного поля на другое».

Boolean37°. Даны координаты двух различных полей шахматной доски x_1, y_1, x_2, y_2 (целые числа, лежащие в диапазоне 1–8). Проверить истинность высказывания: «Король за один ход может перейти с одного поля на другое».

Boolean38°. Даны координаты двух различных полей шахматной доски x_1, y_1, x_2, y_2 (целые числа, лежащие в диапазоне 1–8). Проверить истинность высказывания: «Слон за один ход может перейти с одного поля на другое».

Boolean39°. Даны координаты двух различных полей шахматной доски x_1, y_1, x_2, y_2 (целые числа, лежащие в диапазоне 1–8). Проверить истинность высказывания: «Ферзь за один ход может перейти с одного поля на другое».

Boolean40°. Даны координаты двух различных полей шахматной доски x_1, y_1, x_2, y_2 (целые числа, лежащие в диапазоне 1–8). Проверить истинность высказывания: «Конь за один ход может перейти с одного поля на другое».

Условный оператор

If1. Дано целое число. Если оно является положительным, то прибавить к нему 1; в противном случае не изменять его. Вывести полученное число.

If2. Дано целое число. Если оно является положительным, то прибавить к нему 1; в противном случае вычесть из него 2. Вывести полученное число.

If3. Дано целое число. Если оно является положительным, то прибавить к нему 1; если отрицательным, то вычесть из него 2; если нулевым, то заменить его на 10. Вывести полученное число.

If4°. Даны три целых числа. Найти количество положительных чисел в исходном наборе.

If5. Даны три целых числа. Найти количество положительных и количество отрицательных чисел в исходном наборе.

If6°. Даны два числа. Вывести большее из них.

If7. Даны два числа. Вывести порядковый номер меньшего из них.

If8°. Даны два числа. Вывести вначале большее, а затем меньшее из них.

If9. Даны две переменные вещественного типа: A, B. Перераспределить значения данных переменных так, чтобы в A оказалось меньшее из значений, а в B — большее. Вывести новые значения переменных A и B.

If10. Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной сумму этих значений, а если равны, то присвоить переменным нулевые значения. Вывести новые значения переменных A и B.

If11. Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной большее из этих значений, а если равны, то присвоить переменным нулевые значения. Вывести новые значения переменных A и B.

If12°. Даны три числа. Найти наименьшее из них.

If13. Даны три числа. Найти среднее из них (то есть число, расположенное между наименьшим и наибольшим).

If14. Даны три числа. Вывести вначале наименьшее, а затем наибольшее из данных чисел.

If15. Даны три числа. Найти сумму двух наибольших из них.

If16. Даны три переменные вещественного типа: A, B, C. Если их значения упорядочены по возрастанию, то удвоить их; в противном случае заменить значение каждой переменной на противоположное. Вывести новые

значения переменных A, B, C.

If17. Даны три переменные вещественного типа: A, B, C. Если их значения упорядочены по возрастанию или убыванию, то удвоить их; в противном случае заменить значение каждой переменной на противоположное. Вывести новые значения переменных A, B, C.

If18. Даны три целых числа, одно из которых отлично от двух других, равных между собой. Определить порядковый номер числа, отличного от остальных.

If19. Даны четыре целых числа, одно из которых отлично от трех других, равных между собой. Определить порядковый номер числа, отличного от остальных.

If20. На числовой оси расположены три точки: A, B, C. Определить, какая из двух последних точек (B или C) расположена ближе к A, и вывести эту точку и ее расстояние от точки A.

If21. Даны целочисленные координаты точки на плоскости. Если точка совпадает с началом координат, то вывести 0. Если точка не совпадает с началом координат, но лежит на оси OX или OY, то вывести соответственно 1 или 2. Если точка не лежит на координатных осях, то вывести 3.

If22°. Даны координаты точки, не лежащей на координатных осях OX и OY. Определить номер координатной четверти, в которой находится данная точка.

If23. Даны целочисленные координаты трех вершин прямоугольника, стороны которого параллельны координатным осям. Найти координаты его четвертой вершины.

If24. Для данного вещественного x найти значение следующей функции f, принимающей вещественные значения:

$$f(x) = \begin{cases} 2\cdot\sin(x), & \text{если } x > 0, \\ 6 - x, & \text{если } x \le 0. \end{cases}$$

If25. Для данного целого x найти значение следующей функции f, принимающей значения целого типа:

$$f(x) = \begin{cases} 2\cdot x, & \text{если } x < -2 \text{ или } x > 2, \\ -3\cdot x, & \text{в противном случае.} \end{cases}$$

If26°. Для данного вещественного x найти значение следующей функции f, принимающей вещественные значения:

$$f(x) = \begin{cases} -x, & \text{если } x \le 0, \\ x^2, & \text{если } 0 < x < 2, \\ 4, & \text{если } x \ge 2. \end{cases}$$

If27. Для данного вещественного x найти значение следующей функции f, принимающей значения целого типа:

$$f(x) = \begin{cases} 0, & \text{если } x < 0, \\ 1, & \text{если } x \text{ принадлежит } [0,1), [2,3), \ldots, \\ -1, & \text{если } x \text{ принадлежит } [1,2), [3,4), \ldots. \end{cases}$$

If28. Дан номер года (положительное целое число). Определить количество дней в этом году, учитывая, что обычный год насчитывает 365 дней, а високосный — 366 дней. Високосным считается год, делящийся на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400 (например, годы 300, 1300 и 1900 не являются високосными, а 1200 и 2000 — являются).

If29. Дано целое число. Вывести его строку-описание вида «отрицательное четное число», «нулевое число», «положительное нечетное число» и т. д.

If30. Дано целое число, лежащее в диапазоне 1–999. Вывести его строку-описание вида «четное двузначное число», «нечетное трехзначное число» и т. д.

Оператор выбора

Case1. Дано целое число в диапазоне 1–7. Вывести строку — название дня недели, соответствующее данному числу (1 — «понедельник», 2 — «вторник» и т. д.).

Case2°. Дано целое число K. Вывести строку-описание оценки, соответствующей числу K (1 — «плохо», 2 — «неудовлетворительно», 3 — «удовлетворительно», 4 — «хорошо», 5 — «отлично»). Если K не лежит в диапазоне 1–5, то вывести строку «ошибка».

Case3. Дан номер месяца — целое число в диапазоне 1–12 (1 — январь, 2 — февраль и т. д.). Вывести название соответствующего времени года («зима», «весна», «лето», «осень»).

Case4°. Дан номер месяца — целое число в диапазоне 1–12 (1 — январь, 2 — февраль и т. д.). Определить количество дней в этом месяце для невисокосного года.

Case5. Арифметические действия над числами пронумерованы следующим образом: 1 — сложение, 2 — вычитание, 3 — умножение, 4 — деление. Дан номер действия N (целое число в диапазоне 1–4) и вещественные числа A и B (B не равно 0). Выполнить над числами указанное действие и вывести результат.

Case6. Единицы длины пронумерованы следующим образом: 1 — дециметр, 2 — километр, 3 — метр, 4 — миллиметр, 5 — сантиметр. Дан номер единицы длины (целое число в диапазоне 1–5) и длина отрезка в этих единицах (вещественное число). Найти длину отрезка в метрах.

Case7. Единицы массы пронумерованы следующим образом: 1 — килограмм, 2 — миллиграмм, 3 — грамм, 4 — тонна, 5 — центнер. Дан номер единицы массы (целое число в диапазоне 1–5) и масса тела в этих единицах (вещественное число). Найти массу тела в килограммах.

Case8. Даны два целых числа: D (день) и M (месяц), определяющие правильную дату невисокосного года. Вывести значения D и M для даты, предшествующей указанной.

Case9°. Даны два целых числа: D (день) и M (месяц), определяющие правильную дату невисокосного года. Вывести значения D и M для даты, следующей за указанной.

Case10°. Робот может перемещаться в четырех направлениях («С» — север, «З» — запад, «Ю» — юг, «В» — восток) и принимать три цифровые команды: 0 — продолжать движение, 1 — поворот налево, -1 — поворот направо. Дан символ C — исходное направление робота и целое число N — посланная ему команда. Вывести направление робота после выполнения полученной команды.

Case11. Локатор ориентирован на одну из сторон света («С» — север, «З» — запад, «Ю» — юг, «В» — восток) и может принимать три цифровые команды поворота: 1 — поворот налево, -1 — поворот направо, 2 — поворот на 180°. Дан символ C — исходная ориентация локатора и целые числа N_1 и N_2 — две посланные команды. Вывести ориентацию локатора после выполнения этих команд.

Case12. Элементы окружности пронумерованы следующим образом: 1 — радиус R, 2 — диаметр $D = 2{\cdot}R$, 3 — длина $L = 2{\cdot}\pi{\cdot}R$, 4 — площадь круга $S = \pi{\cdot}R^2$. Дан номер одного из этих элементов и его значение. Вывести значения остальных элементов данной окружности (в том же порядке). В качестве значения π использовать 3.14.

Case13. Элементы равнобедренного прямоугольного треугольника пронумерованы следующим образом: 1 — катет a, 2 — гипотенуза $c = a \cdot \sqrt{2}$, 3 — высота h, опущенная на гипотенузу ($h = c/2$), 4 — площадь $S = c \cdot h/2$. Дан номер одного из этих элементов и его значение. Вывести значения остальных элементов данного треугольника (в том же порядке).

Case14. Элементы равностороннего треугольника пронумерованы следующим образом: 1 — сторона a, 2 — радиус R_1 вписанной окружности ($R_1 = a \cdot \sqrt{3}/6$), 3 — радиус R_2 описанной окружности ($R_2 = 2 \cdot R_1$), 4 — площадь $S = a^2 \cdot \sqrt{3}/4$. Дан номер одного из этих элементов и его значение. Вывести значения остальных элементов данного треугольника (в том же порядке).

Case15. Мастям игральных карт присвоены порядковые номера: 1 — пики, 2 — трефы, 3 — бубны, 4 — червы. Достоинству карт, старших десятки, присвоены номера: 11 — валет, 12 — дама, 13 — король, 14 — туз. Даны два целых числа: N — достоинство ($6 \le N \le 14$) и M — масть карты ($1 \le M \le 4$). Вывести название соответствующей карты вида «шестерка бубен», «дама червей», «туз треф» и т. п.

Case16. Дано целое число в диапазоне 20–69, определяющее возраст (в годах). Вывести строку-описание указанного возраста, обеспечив правильное согласование числа со словом «год», например: 20 — «двадцать лет», 32 — «тридцать два года», 41 — «сорок один год».

Case17. Дано целое число в диапазоне 10–40, определяющее количество учебных заданий по некоторой теме. Вывести строку-описание указанного количества заданий, обеспечив правильное согласование числа со словами «учебное задание», например: 18 — «восемнадцать учебных заданий», 23 — «двадцать три учебных задания», 31 — «тридцать одно учебное задание».

Case18°. Дано целое число в диапазоне 100–999. Вывести строку-описание данного числа, например: 256 — «двести пятьдесят шесть», 814 — «восемьсот четырнадцать».

Case19. В восточном календаре принят 60-летний цикл, состоящий из 12-летних подциклов, обозначаемых названиями цвета: зеленый, красный, желтый, белый и черный. В каждом подцикле годы носят названия животных: крысы, коровы, тигра, зайца, дракона, змеи, лошади, овцы, обезьяны, курицы, собаки и свиньи. По номеру года определить его название, если 1984 год — начало цикла: «год зеленой крысы».

Case20. Даны два целых числа: D (день) и M (месяц), определяющие правильную дату. Вывести знак Зодиака, соответствующий этой дате: «Водолей» (20.1–18.2), «Рыбы» (19.2–20.3), «Овен» (21.3–19.4), «Телец» (20.4–20.5), «Близнецы» (21.5–21.6), «Рак» (22.6–22.7), «Лев» (23.7–22.8), «Дева» (23.8–22.9), «Весы» (23.9–22.10), «Скорпион» (23.10–22.11), «Стрелец» (23.11–21.12), «Козерог» (22.12–19.1).

Цикл с параметром

For1. Даны целые числа K и N ($N > 0$). Вывести N раз число K.

For2. Даны два целых числа A и B ($A < B$). Вывести в порядке возрастания все целые числа, расположенные между A и B (включая сами числа A и B), а также количество N этих чисел.

For3. Даны два целых числа A и B ($A < B$). Вывести в порядке убывания все целые числа, расположенные между A и B (не включая числа A и B), а также количество N этих чисел.

For4. Дано вещественное число — цена 1 кг конфет. Вывести стоимость 1, 2, . . . , 10 кг конфет.

For5°. Дано вещественное число — цена 1 кг конфет. Вывести стоимость 0.1, 0.2, . . . , 1 кг конфет.

For6. Дано вещественное число — цена 1 кг конфет. Вывести стоимость 1.2, 1.4, . . . , 2 кг конфет.

For7. Даны два целых числа A и B ($A < B$). Найти сумму всех целых чисел от A до B включительно.

For8. Даны два целых числа A и B ($A < B$). Найти произведение всех целых чисел от A до B включительно.

For9. Даны два целых числа A и B ($A < B$). Найти сумму квадратов всех целых чисел от A до B включительно.

For10. Дано целое число N (> 0). Найти сумму

$$1 + 1/2 + 1/3 + \ldots + 1/N$$

(вещественное число).

For11. Дано целое число N (> 0). Найти сумму

$$N^2 + (N+1)^2 + (N+2)^2 + \ldots + (2 \cdot N)^2$$

(целое число).

For12°. Дано целое число N (> 0). Найти произведение

$$1.1 \cdot 1.2 \cdot 1.3 \cdot \ldots$$

(N сомножителей).

For13°. Дано целое число N (>0). Найти значение выражения

$$1.1 - 1.2 + 1.3 - \ldots$$

(N слагаемых, знаки чередуются). Условный оператор не использовать.

For14. Дано целое число N (>0). Найти квадрат данного числа, используя для его вычисления следующую формулу:

$$N^2 = 1 + 3 + 5 + \ldots + (2 \cdot N - 1).$$

После добавления к сумме каждого слагаемого выводить текущее значение суммы (в результате будут выведены квадраты всех целых чисел от 1 до N).

For15°. Дано вещественное число A и целое число N (>0). Найти A в степени N:

$$A^N = A \cdot A \cdot \ldots \cdot A$$

(числа A перемножаются N раз).

For16°. Дано вещественное число A и целое число N (>0). Используя один цикл, вывести все целые степени числа A от 1 до N.

For17. Дано вещественное число A и целое число N (>0). Используя один цикл, найти сумму

$$1 + A + A^2 + A^3 + \ldots + A^N.$$

For18. Дано вещественное число A и целое число N (>0). Используя один цикл, найти значение выражения

$$1 - A + A^2 - A^3 + \ldots + (-1)^N \cdot A^N.$$

Условный оператор не использовать.

For19°. Дано целое число N (>0). Найти произведение

$$N! = 1 \cdot 2 \cdot \ldots \cdot N$$

(*N–факториал*). Чтобы избежать целочисленного переполнения, вычислять это произведение с помощью вещественной переменной и вывести его как вещественное число.

For20°. Дано целое число N (>0). Используя один цикл, найти сумму

$$1! + 2! + 3! + \ldots + N!$$

(выражение $N!$ — *N–факториал* — обозначает произведение всех целых чисел от 1 до N: $N! = 1 \cdot 2 \cdot \ldots \cdot N$). Чтобы избежать целочисленного переполнения, проводить вычисления с помощью вещественных переменных и вывести результат как вещественное число.

For21. Дано целое число N (>0). Используя один цикл, найти сумму

$$1 + 1/(1!) + 1/(2!) + 1/(3!) + \ldots + 1/(N!)$$

(выражение $N!$ — *N–факториал* — обозначает произведение всех целых чисел от 1 до N: $N! = 1 \cdot 2 \cdot \ldots \cdot N$). Полученное число является приближенным значением константы $e = \exp(1)$.

For22. Дано вещественное число X и целое число N (>0). Найти значение выражения

$$1 + X + X^2/(2!) + \ldots + X^N/(N!)$$

($N! = 1 \cdot 2 \cdot \ldots \cdot N$). Полученное число является приближенным значением функции exp в точке X.

For23. Дано вещественное число X и целое число N (>0). Найти значение выражения

$$X - X^3/(3!) + X^5/(5!) - \ldots + (-1)^N \cdot X^{2 \cdot N+1}/((2 \cdot N+1)!)$$

($N! = 1 \cdot 2 \cdot \ldots \cdot N$). Полученное число является приближенным значением функции sin в точке X.

For24. Дано вещественное число X и целое число N (>0). Найти значение выражения

$$1 - X^2/(2!) + X^4/(4!) - \ldots + (-1)^N \cdot X^{2 \cdot N}/((2 \cdot N)!)$$

($N! = 1 \cdot 2 \cdot \ldots \cdot N$). Полученное число является приближенным значением функции cos в точке X.

For25. Дано вещественное число X ($|X|<1$) и целое число N (>0). Найти значение выражения

$$X - X^2/2 + X^3/3 - \ldots + (-1)^{N-1} \cdot X^N/N.$$

Полученное число является приближенным значением функции ln в точке $1+X$.

For26. Дано вещественное число X ($|X|<1$) и целое число N (>0). Найти значение выражения

$$X - X^3/3 + X^5/5 - \ldots + (-1)^N \cdot X^{2 \cdot N+1}/(2 \cdot N+1).$$

Полученное число является приближенным значением функции arctg в точке X.

For27. Дано вещественное число X ($|X|<1$) и целое число N (>0). Найти значение выражения

$$X + 1 \cdot X^3/(2 \cdot 3) + 1 \cdot 3 \cdot X^5/(2 \cdot 4 \cdot 5) + \ldots +$$
$$+ 1 \cdot 3 \cdot \ldots \cdot (2 \cdot N-1) \cdot X^{2 \cdot N+1}/(2 \cdot 4 \cdot \ldots \cdot (2 \cdot N) \cdot (2 \cdot N+1)).$$

Полученное число является приближенным значением функции arcsin в точке X.

For28. Дано вещественное число X ($|X|<1$) и целое число N (>0). Найти значение выражения

$$1 + X/2 - 1\cdot X^2/(2\cdot 4) + 1\cdot 3\cdot X^3/(2\cdot 4\cdot 6) - \ldots +$$
$$+ (-1)^{N-1}\cdot 1\cdot 3\cdot\ldots\cdot(2\cdot N-3)\cdot X^N/(2\cdot 4\cdot\ldots\cdot(2\cdot N)).$$

Полученное число является приближенным значением функции $\sqrt{1+X}$.

For29. Дано целое число N (>1) и две вещественные точки на числовой оси: A, B ($A < B$). Отрезок $[A, B]$ разбит на N равных отрезков. Вывести H — длину каждого отрезка, а также набор точек

$$A, \quad A+H, \quad A+2\cdot H, \quad A+3\cdot H, \quad \ldots, \quad B,$$

образующий разбиение отрезка $[A, B]$.

For30. Дано целое число N (>1) и две вещественные точки на числовой оси: A, B ($A < B$). Отрезок $[A, B]$ разбит на N равных отрезков. Вывести H — длину каждого отрезка, а также значения функции $F(X) = 1 - \sin(X)$ в точках, разбивающих отрезок $[A, B]$:

$$F(A), \quad F(A+H), \quad F(A+2\cdot H), \quad \ldots, \quad F(B).$$

For31. Дано целое число N (>0). Последовательность вещественных чисел A_K определяется следующим образом:

$$A_0 = 2, \qquad A_K = 2 + 1/A_{K-1}, \quad K = 1, 2, \ldots .$$

Вывести элементы $A_1, A_2, \ldots, A_N$.

For32. Дано целое число N (>0). Последовательность вещественных чисел A_K определяется следующим образом:

$$A_0 = 1, \qquad A_K = (A_{K-1} + 1)/K, \quad K = 1, 2, \ldots .$$

Вывести элементы $A_1, A_2, \ldots, A_N$.

For33°. Дано целое число N (>1). Последовательность *чисел Фибоначчи* F_K (целого типа) определяется следующим образом:

$$F_1 = 1, \qquad F_2 = 1, \qquad F_K = F_{K-2} + F_{K-1}, \quad K = 3, 4, \ldots .$$

Вывести элементы F_1, F_2, ..., F_N.

For34. Дано целое число N (>1). Последовательность вещественных чисел A_K определяется следующим образом:

$$A_1 = 1, \qquad A_2 = 2, \qquad A_K = (A_{K-2} + 2\cdot A_{K-1})/3, \quad K = 3, 4, \ldots .$$

Вывести элементы $A_1, A_2, \ldots, A_N$.

For35. Дано целое число N (>2). Последовательность целых чисел A_K определяется следующим образом:

$$A_1 = 1, \qquad A_2 = 2, \qquad A_3 = 3,$$
$$A_K = A_{K-1} + A_{K-2} - 2\cdot A_{K-3}, \quad K = 4, 5, \ldots .$$

Вывести элементы $A_1, A_2, \ldots, A_N$.

Вложенные циклы

For36°. Даны целые положительные числа N и K. Найти сумму

$$1^K + 2^K + \ldots + N^K.$$

Чтобы избежать целочисленного переполнения, вычислять слагаемые этой суммы с помощью вещественной переменной и выводить результат как вещественное число.

For37. Дано целое число N (>0). Найти сумму

$$1^1 + 2^2 + \ldots + N^N.$$

Чтобы избежать целочисленного переполнения, вычислять слагаемые этой суммы с помощью вещественной переменной и выводить результат как вещественное число.

For38. Дано целое число N (>0). Найти сумму

$$1^N + 2^{N-1} + \ldots + N^1.$$

Чтобы избежать целочисленного переполнения, вычислять слагаемые этой суммы с помощью вещественной переменной и выводить результат как вещественное число.

For39. Даны целые положительные числа A и B ($A < B$). Вывести все целые числа от A до B включительно; при этом каждое число должно выводиться столько раз, каково его значение (например, число 3 выводится 3 раза).

For40. Даны целые числа A и B ($A < B$). Вывести все целые числа от A до B включительно; при этом число A должно выводиться 1 раз, число $A + 1$ должно выводиться 2 раза и т. д.

Цикл с условием

While1°. Даны положительные числа A и B ($A > B$). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Не используя операции умножения и деления, найти длину незанятой части отрезка A.

While2°. Даны положительные числа A и B ($A > B$). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Не используя операции умножения и деления, найти количество отрезков B, размещенных на отрезке A.

While3. Даны целые положительные числа N и K. Используя только операции сложения и вычитания, найти частное от деления нацело N на K, а также

остаток от этого деления.

While4°. Дано целое число N (>0). Если оно является степенью числа 3, то вывести TRUE, если не является — вывести FALSE.

While5. Дано целое число N (>0), являющееся некоторой степенью числа 2: $N = 2^K$. Найти целое число K — показатель этой степени.

While6. Дано целое число N (>0). Найти *двойной факториал* N:

$$N!! = N\cdot(N-2)\cdot(N-4)\cdot\ldots$$

(последний сомножитель равен 2, если N — четное, и 1, если N — нечетное). Чтобы избежать целочисленного переполнения, вычислять это произведение с помощью вещественной переменной и вывести его как вещественное число.

While7°. Дано целое число N (>0). Найти наименьшее целое положительное число K, квадрат которого превосходит N: $K^2 > N$. Функцию извлечения квадратного корня не использовать.

While8. Дано целое число N (>0). Найти наибольшее целое число K, квадрат которого не превосходит N: $K^2 \le N$. Функцию извлечения квадратного корня не использовать.

While9. Дано целое число N (>1). Найти наименьшее целое число K, при котором выполняется неравенство $3^K > N$.

While10. Дано целое число N (>1). Найти наибольшее целое число K, при котором выполняется неравенство $3^K < N$.

While11°. Дано целое число N (>1). Вывести наименьшее из целых чисел K, для которых сумма $1 + 2 + \ldots + K$ будет больше или равна N, и саму эту сумму.

While12°. Дано целое число N (>1). Вывести наибольшее из целых чисел K, для которых сумма $1 + 2 + \ldots + K$ будет меньше или равна N, и саму эту сумму.

While13. Дано число A (>1). Вывести наименьшее из целых чисел K, для которых сумма $1 + 1/2 + \ldots + 1/K$ будет больше A, и саму эту сумму.

While14. Дано число A (>1). Вывести наибольшее из целых чисел K, для которых сумма $1 + 1/2 + \ldots + 1/K$ будет меньше A, и саму эту сумму.

While15. Начальный вклад в банке равен 1000 руб. Через каждый месяц размер вклада увеличивается на P процентов от имеющейся суммы (P — вещественное число, $0<P<25$). По данному P определить, через сколько месяцев размер вклада превысит 1100 руб., и вывести найденное количество месяцев K (целое число) и итоговый размер вклада S (вещественное

число).

While16. Спортсмен-лыжник начал тренировки, пробежав в первый день 10 км. Каждый следующий день он увеличивал длину пробега на P процентов от пробега предыдущего дня (P — вещественное, $0 < P < 50$). По данному P определить, после какого дня суммарный пробег лыжника за все дни превысит 200 км, и вывести найденное количество дней K (целое) и суммарный пробег S (вещественное число).

While17. Дано целое число N (> 0). Используя операции деления нацело и взятия остатка от деления, вывести все его цифры, начиная с самой правой (разряда единиц).

While18. Дано целое число N (> 0). Используя операции деления нацело и взятия остатка от деления, найти количество и сумму его цифр.

While19. Дано целое число N (> 0). Используя операции деления нацело и взятия остатка от деления, найти число, полученное при прочтении числа N справа налево.

While20. Дано целое число N (> 0). С помощью операций деления нацело и взятия остатка от деления определить, имеется ли в записи числа N цифра «2». Если имеется, то вывести TRUE, если нет — вывести FALSE.

While21. Дано целое число N (> 0). С помощью операций деления нацело и взятия остатка от деления определить, имеются ли в записи числа N нечетные цифры. Если имеются, то вывести TRUE, если нет — вывести FALSE.

While22°. Дано целое число N (> 1). Если оно является *простым*, то есть не имеет положительных делителей, кроме 1 и самого себя, то вывести TRUE, иначе вывести FALSE.

While23°. Даны целые положительные числа A и B. Найти их *наибольший общий делитель* (НОД), используя *алгоритм Евклида*:

$$\text{НОД}(A, B) = \text{НОД}(B, A \bmod B), \quad \text{если } B \neq 0; \qquad \text{НОД}(A, 0) = A,$$

где «mod» обозначает операцию взятия остатка от деления.

While24. Дано целое число N (> 1). Последовательность *чисел Фибоначчи* F_K определяется следующим образом:

$$F_1 = 1, \qquad F_2 = 1, \qquad F_K = F_{K-2} + F_{K-1}, \quad K = 3, 4, \ldots .$$

Проверить, является ли число N числом Фибоначчи. Если является, то вывести TRUE, если нет — вывести FALSE.

While25. Дано целое число N (> 1). Найти первое число Фибоначчи, большее N (определение *чисел Фибоначчи* дано в задании While24).

While26. Дано целое число N (>1), являющееся числом Фибоначчи: $N = F_K$ (определение *чисел Фибоначчи* дано в задании While24). Найти целые числа F_{K-1} и F_{K+1} — предыдущее и последующее числа Фибоначчи.

While27. Дано целое число N (>1), являющееся числом Фибоначчи: $N = F_K$ (определение *чисел Фибоначчи* дано в задании While24). Найти целое число K — порядковый номер числа Фибоначчи N.

While28. Дано вещественное число ε (>0). Последовательность вещественных чисел A_K определяется следующим образом:

$$A_1 = 2, \qquad A_K = 2 + 1/A_{K-1}, \quad K = 2, 3, \ldots .$$

Найти первый из номеров K, для которых выполняется условие $|A_K - A_{K-1}| < \varepsilon$, и вывести этот номер, а также числа A_{K-1} и A_K.

While29. Дано вещественное число ε (>0). Последовательность вещественных чисел A_K определяется следующим образом:

$$A_1 = 1, \qquad A_2 = 2, \qquad A_K = (A_{K-2} + 2 \cdot A_{K-1})/3, \quad K = 3, 4, \ldots .$$

Найти первый из номеров K, для которых выполняется условие $|A_K - A_{K-1}| < \varepsilon$, и вывести этот номер, а также числа A_{K-1} и A_K.

While30. Даны положительные числа A, B, C. На прямоугольнике размера $A \times B$ размещено максимально возможное количество квадратов со стороной C (без наложений). Найти количество квадратов, размещенных на прямоугольнике. Операции умножения и деления не использовать.

Последовательности

Во всех заданиях данной группы предполагается, что исходный набор содержит ненулевое число элементов (в частности, число N всегда больше нуля). В заданиях на обработку нескольких наборов чисел (Series29–Series40) количество наборов K также всегда является ненулевым.

Series1°. Даны десять вещественных чисел. Найти их сумму.

Series2. Даны десять вещественных чисел. Найти их произведение.

Series3. Даны десять вещественных чисел. Найти их среднее арифметическое.

Series4. Дано целое число N и набор из N вещественных чисел. Вывести сумму и произведение чисел из данного набора.

Series5. Дано целое число N и набор из N положительных вещественных чисел. Вывести в том же порядке целые части всех чисел из данного набора (как вещественные числа с нулевой дробной частью), а также сумму всех целых частей.

Series6. Дано целое число N и набор из N положительных вещественных чисел. Вывести в том же порядке дробные части всех чисел из данного набора (как вещественные числа с нулевой целой частью), а также произведение всех дробных частей.

Series7. Дано целое число N и набор из N вещественных чисел. Вывести в том же порядке округленные значения всех чисел из данного набора (как целые числа), а также сумму всех округленных значений.

Series8. Дано целое число N и набор из N целых чисел. Вывести в том же порядке все четные числа из данного набора и количество K таких чисел.

Series9. Дано целое число N и набор из N целых чисел. Вывести в том же порядке номера всех нечетных чисел из данного набора и количество K таких чисел.

Series10. Дано целое число N и набор из N целых чисел. Если в наборе имеются положительные числа, то вывести TRUE; в противном случае вывести FALSE.

Series11. Даны целые числа K, N и набор из N целых чисел. Если в наборе имеются числа, меньшие K, то вывести TRUE; в противном случае вывести FALSE.

Series12. Дан набор ненулевых целых чисел; признак его завершения — число 0. Вывести количество чисел в наборе.

Series13. Дан набор ненулевых целых чисел; признак его завершения — число 0. Вывести сумму всех положительных четных чисел из данного набора. Если требуемые числа в наборе отсутствуют, то вывести 0.

Series14. Дано целое число K и набор ненулевых целых чисел; признак его завершения — число 0. Вывести количество чисел в наборе, меньших K.

Series15°. Дано целое число K и набор ненулевых целых чисел; признак его завершения — число 0. Вывести номер первого числа в наборе, большего K. Если таких чисел нет, то вывести 0.

Series16°. Дано целое число K и набор ненулевых целых чисел; признак его завершения — число 0. Вывести номер последнего числа в наборе, большего K. Если таких чисел нет, то вывести 0.

Series17°. Дано вещественное число B, целое число N и набор из N вещественных чисел, упорядоченных по возрастанию. Вывести элементы набора вместе с числом B, сохраняя упорядоченность выводимых чисел.

Series18. Дано целое число N и набор из N целых чисел, упорядоченный по возрастанию. Данный набор может содержать одинаковые элементы.

Вывести в том же порядке все различные элементы данного набора.

Series19°. Дано целое число N (>1) и набор из N целых чисел. Вывести те элементы в наборе, которые меньше своего левого соседа, и количество K таких элементов.

Series20. Дано целое число N (>1) и набор из N целых чисел. Вывести те элементы в наборе, которые меньше своего правого соседа, и количество K таких элементов.

Series21°. Дано целое число N (>1) и набор из N вещественных чисел. Проверить, образует ли данный набор возрастающую последовательность. Если образует, то вывести TRUE, если нет — вывести FALSE.

Series22. Дано целое число N (>1) и набор из N вещественных чисел. Если данный набор образует убывающую последовательность, то вывести 0; в противном случае вывести номер первого числа, нарушающего закономерность.

Series23. Дано целое число N (>2) и набор из N вещественных чисел. Набор называется *пилообразным*, если каждый его внутренний элемент либо больше, либо меньше обоих своих соседей (то есть является «зубцом»). Если данный набор является пилообразным, то вывести 0; в противном случае вывести номер первого элемента, не являющегося зубцом.

Series24. Дано целое число N и набор из N целых чисел, содержащий по крайней мере два нуля. Вывести сумму чисел из данного набора, расположенных между последними двумя нулями (если последние нули идут подряд, то вывести 0).

Series25. Дано целое число N и набор из N целых чисел, содержащий по крайней мере два нуля. Вывести сумму чисел из данного набора, расположенных между первым и последним нулем (если первый и последний нули идут подряд, то вывести 0).

Вложенные циклы

Series26. Даны целые числа K, N и набор из N вещественных чисел: A_1, A_2, ..., A_N. Вывести K-е степени чисел из данного набора:

$$(A_1)^K,\ (A_2)^K,\ \ldots,\ (A_N)^K.$$

Series27. Дано целое число N и набор из N вещественных чисел: A_1, A_2, ..., A_N. Вывести следующие числа:

$$A_1,\ (A_2)^2,\ \ldots,\ (A_{N-1})^{N-1},\ (A_N)^N.$$

Series28. Дано целое число N и набор из N вещественных чисел: A_1, A_2, ..., A_N. Вывести следующие числа:

$$(A_1)^N,\ (A_2)^{N-1},\ \ldots,\ (A_{N-1})^2,\ A_N.$$

Series29. Даны целые числа K, N, а также K наборов целых чисел по N элементов в каждом наборе. Вывести общую сумму всех элементов, входящих в данные наборы.

Series30°. Даны целые числа K, N, а также K наборов целых чисел по N элементов в каждом наборе. Для каждого набора вывести сумму его элементов.

Series31. Даны целые числа K, N, а также K наборов целых чисел по N элементов в каждом наборе. Найти количество наборов, содержащих число 2. Если таких наборов нет, то вывести 0.

Series32. Даны целые числа K, N, а также K наборов целых чисел по N элементов в каждом наборе. Для каждого набора вывести номер его первого элемента, равного 2, или число 0, если в данном наборе нет двоек.

Series33. Даны целые числа K, N, а также K наборов целых чисел по N элементов в каждом наборе. Для каждого набора вывести номер его последнего элемента, равного 2, или число 0, если в данном наборе нет двоек.

Series34. Даны целые числа K, N, а также K наборов целых чисел по N элементов в каждом наборе. Для каждого набора выполнить следующее действие: если в наборе содержится число 2, то вывести сумму его элементов; если в наборе нет двоек, то вывести 0.

Series35. Дано целое число K, а также K наборов ненулевых целых чисел. Признаком завершения каждого набора является число 0. Для каждого набора вывести количество его элементов. Вывести также общее количество элементов во всех наборах.

Series36. Дано целое число K, а также K наборов ненулевых целых чисел. Каждый набор содержит не менее двух элементов, признаком его завершения является число 0. Найти количество наборов, элементы которых возрастают.

Series37. Дано целое число K, а также K наборов ненулевых целых чисел. Каждый набор содержит не менее двух элементов, признаком его завершения является число 0. Найти количество наборов, элементы которых возрастают или убывают.

Series38. Дано целое число K, а также K наборов ненулевых целых чисел. Каждый набор содержит не менее двух элементов, признаком его завер-

шения является число 0. Для каждого набора выполнить следующее действие: если элементы набора возрастают, то вывести 1; если элементы набора убывают, то вывести -1; если элементы набора не возрастают и не убывают, то вывести 0.

Series39. Дано целое число K, а также K наборов ненулевых целых чисел. Каждый набор содержит не менее трех элементов, признаком его завершения является число 0. Найти количество пилообразных наборов (определение пилообразного набора дано в задании Series23).

Series40. Дано целое число K, а также K наборов ненулевых целых чисел. Каждый набор содержит не менее трех элементов, признаком его завершения является число 0. Для каждого набора выполнить следующее действие: если набор является пилообразным (см. задание Series23), то вывести количество его элементов; в противном случае вывести номер первого элемента, который не является зубцом.

Процедуры и функции

Процедуры с числовыми параметрами

Proc1. Описать процедуру PowerA3(A, B), вычисляющую третью степень числа A и возвращающую ее в переменной B (A — входной, B — выходной параметр; оба параметра являются вещественными). С помощью этой процедуры найти третьи степени пяти данных чисел.

Proc2. Описать процедуру PowerA234(A, B, C, D), вычисляющую вторую, третью и четвертую степень числа A и возвращающую эти степени соответственно в переменных B, C и D (A — входной, B, C, D — выходные параметры; все параметры являются вещественными). С помощью этой процедуры найти вторую, третью и четвертую степень пяти данных чисел.

Proc3. Описать процедуру Mean(X, Y, $AMean$, $GMean$), вычисляющую *среднее арифметическое* $AMean = (X+Y)/2$ и *среднее геометрическое* $GMean = \sqrt{X \cdot Y}$ двух положительных чисел X и Y (X и Y — входные, $AMean$ и $GMean$ — выходные параметры вещественного типа). С помощью этой процедуры найти среднее арифметическое и среднее геометрическое для пар (A, B), (A, C), (A, D), если даны A, B, C, D.

Proc4°. Описать процедуру TrianglePS(a, P, S), вычисляющую по стороне a равностороннего треугольника его периметр $P = 3 \cdot a$ и площадь $S = a^2 \cdot \sqrt{3}/4$

(a — входной, P и S — выходные параметры; все параметры являются вещественными). С помощью этой процедуры найти периметры и площади трех равносторонних треугольников с данными сторонами.

Proc5. Описать процедуру RectPS(x_1, y_1, x_2, y_2, P, S), вычисляющую периметр P и площадь S прямоугольника со сторонами, параллельными осям координат, по координатам (x_1, y_1), (x_2, y_2) его противоположных вершин (x_1, y_1, x_2, y_2 — входные, P и S — выходные параметры вещественного типа). С помощью этой процедуры найти периметры и площади трех прямоугольников с данными противоположными вершинами.

Proc6. Описать процедуру DigitCountSum(K, C, S), находящую количество C цифр целого положительного числа K, а также их сумму S (K — входной, C и S — выходные параметры целого типа). С помощью этой процедуры найти количество и сумму цифр для каждого из пяти данных целых чисел.

Proc7. Описать процедуру InvertDigits(K), меняющую порядок следования цифр целого положительного числа K на обратный (K — параметр целого типа, являющийся одновременно входным и выходным). С помощью этой процедуры поменять порядок следования цифр на обратный для каждого из пяти данных целых чисел.

Proc8°. Описать процедуру AddRightDigit(D, K), добавляющую к целому положительному числу K справа цифру D (D — входной параметр целого типа, лежащий в диапазоне 0–9, K — параметр целого типа, являющийся одновременно входным и выходным). С помощью этой процедуры последовательно добавить к данному числу K справа данные цифры D_1 и D_2, выводя результат каждого добавления.

Proc9. Описать процедуру AddLeftDigit(D, K), добавляющую к целому положительному числу K слева цифру D (D — входной параметр целого типа, лежащий в диапазоне 1–9, K — параметр целого типа, являющийся одновременно входным и выходным). С помощью этой процедуры последовательно добавить к данному числу K слева данные цифры D_1 и D_2, выводя результат каждого добавления.

Proc10°. Описать процедуру Swap(X, Y), меняющую содержимое переменных X и Y (X и Y — вещественные параметры, являющиеся одновременно входными и выходными). С ее помощью для данных переменных A, B, C, D последовательно поменять содержимое следующих пар: A и B, C и D, B и C и вывести новые значения A, B, C, D.

Proc11. Описать процедуру Minmax(X, Y), записывающую в переменную X

минимальное из значений X и Y, а в переменную Y — максимальное из этих значений (X и Y — вещественные параметры, являющиеся одновременно входными и выходными). Используя четыре вызова этой процедуры, найти минимальное и максимальное из данных чисел A, B, C, D.

Proc12. Описать процедуру SortInc3(A, B, C), меняющую содержимое переменных A, B, C таким образом, чтобы их значения оказались упорядоченными по возрастанию (A, B, C — вещественные параметры, являющиеся одновременно входными и выходными). С помощью этой процедуры упорядочить по возрастанию два данных набора из трех чисел: (A_1, B_1, C_1) и (A_2, B_2, C_2).

Proc13. Описать процедуру SortDec3(A, B, C), меняющую содержимое переменных A, B, C таким образом, чтобы их значения оказались упорядоченными по убыванию (A, B, C — вещественные параметры, являющиеся одновременно входными и выходными). С помощью этой процедуры упорядочить по убыванию два данных набора из трех чисел: (A_1, B_1, C_1) и (A_2, B_2, C_2).

Proc14. Описать процедуру ShiftRight3(A, B, C), выполняющую *правый циклический сдвиг*: значение A переходит в B, значение B — в C, значение C — в A (A, B, C — вещественные параметры, являющиеся одновременно входными и выходными). С помощью этой процедуры выполнить правый циклический сдвиг для двух данных наборов из трех чисел: (A_1, B_1, C_1) и (A_2, B_2, C_2).

Proc15. Описать процедуру ShiftLeft3(A, B, C), выполняющую *левый циклический сдвиг*: значение A переходит в C, значение C — в B, значение B — в A (A, B, C — вещественные параметры, являющиеся одновременно входными и выходными). С помощью этой процедуры выполнить левый циклический сдвиг для двух данных наборов из трех чисел: (A_1, B_1, C_1) и (A_2, B_2, C_2).

Функции с числовыми параметрами

Proc16. Описать функцию Sign(X) целого типа, возвращающую для вещественного числа X следующие значения:

$$-1, \quad \text{если } X < 0; \qquad 0, \quad \text{если } X = 0; \qquad 1, \quad \text{если } X > 0.$$

С помощью этой функции найти значение выражения Sign(A) + Sign(B) для данных вещественных чисел A и B.

Proc17. Описать функцию RootsCount(A, B, C) целого типа, определяющую количество корней квадратного уравнения $A \cdot x^2 + B \cdot x + C = 0$ (A, B, C — вещественные параметры, $A \neq 0$). С ее помощью найти количество корней для каждого из трех квадратных уравнений с данными коэффициентами. Количество корней определять по значению *дискриминанта*:

$$D = B^2 - 4 \cdot A \cdot C.$$

Proc18. Описать функцию CircleS(R) вещественного типа, находящую площадь круга радиуса R (R — вещественное). С помощью этой функции найти площади трех кругов с данными радиусами. Площадь круга радиуса R вычисляется по формуле $S = \pi \cdot R^2$. В качестве значения π использовать 3.14.

Proc19. Описать функцию RingS(R_1, R_2) вещественного типа, находящую площадь кольца, заключенного между двумя окружностями с общим центром и радиусами R_1 и R_2 (R_1 и R_2 — вещественные, $R_1 > R_2$). С ее помощью найти площади трех колец, для которых даны внешние и внутренние радиусы. Воспользоваться формулой площади круга радиуса R: $S = \pi \cdot R^2$. В качестве значения π использовать 3.14.

Proc20°. Описать функцию TriangleP(a, h), находящую периметр равнобедренного треугольника по его основанию a и высоте h, проведенной к основанию (a и h — вещественные). С помощью этой функции найти периметры трех треугольников, для которых даны основания и высоты. Для нахождения боковой стороны b треугольника использовать *теорему Пифагора*:

$$b^2 = (a/2)^2 + h^2.$$

Proc21°. Описать функцию SumRange(A, B) целого типа, находящую сумму всех целых чисел от A до B включительно (A и B — целые). Если $A > B$, то функция возвращает 0. С помощью этой функции найти суммы чисел от A до B и от B до C, если даны числа A, B, C.

Proc22. Описать функцию Calc(A, B, Op) вещественного типа, выполняющую над ненулевыми вещественными числами A и B одну из арифметических операций и возвращающую ее результат. Вид операции определяется целым параметром Op: 1 — вычитание, 2 — умножение, 3 — деление, остальные значения — сложение. С помощью Calc выполнить для данных A и B операции, определяемые данными целыми N_1, N_2, N_3.

Proc23. Описать функцию Quarter(x, y) целого типа, определяющую номер координатной четверти, в которой находится точка с ненулевыми веще-

ственными координатами (x, y). С помощью этой функции найти номера координатных четвертей для трех точек с данными ненулевыми координатами.

Proc24. Описать функцию Even(K) логического типа, возвращающую TRUE, если целый параметр K является четным, и FALSE в противном случае. С ее помощью найти количество четных чисел в наборе из 10 целых чисел.

Proc25°. Описать функцию IsSquare(K) логического типа, возвращающую TRUE, если целый параметр K (>0) является квадратом некоторого целого числа, и FALSE в противном случае. С ее помощью найти количество квадратов в наборе из 10 целых положительных чисел.

Proc26. Описать функцию IsPower5(K) логического типа, возвращающую TRUE, если целый параметр K (>0) является степенью числа 5, и FALSE в противном случае. С ее помощью найти количество степеней числа 5 в наборе из 10 целых положительных чисел.

Proc27. Описать функцию IsPowerN(K, N) логического типа, возвращающую TRUE, если целый параметр K (>0) является степенью числа N (>1), и FALSE в противном случае. Дано число N (>1) и набор из 10 целых положительных чисел. С помощью функции IsPowerN найти количество степеней числа N в данном наборе.

Proc28. Описать функцию IsPrime(N) логического типа, возвращающую TRUE, если целый параметр N (>1) является простым числом, и FALSE в противном случае (число, большее 1, называется *простым*, если оно не имеет положительных делителей, кроме 1 и самого себя). Дан набор из 10 целых чисел, больших 1. С помощью функции IsPrime найти количество простых чисел в данном наборе.

Proc29. Описать функцию DigitCount(K) целого типа, находящую количество цифр целого положительного числа K. Используя эту функцию, найти количество цифр для каждого из пяти данных целых положительных чисел.

Proc30. Описать функцию DigitN(K, N) целого типа, возвращающую N-ю цифру целого положительного числа K (цифры в числе нумеруются справа налево). Если количество цифр в числе K меньше N, то функция возвращает -1. Для каждого из пяти данных целых положительных чисел K_1, K_2, ..., K_5 вызвать функцию DigitN с параметром N, изменяющимся от 1 до 5.

Proc31. Описать функцию IsPalindrom(K), возвращающую TRUE, если целый параметр K (>0) является *палиндромом* (то есть его запись читается оди-

наково слева направо и справа налево), и FALSE в противном случае. С ее помощью найти количество палиндромов в наборе из 10 целых положительных чисел.

Proc32. Описать функцию DegToRad(D) вещественного типа, находящую величину угла в радианах, если дана его величина D в градусах (D — вещественное число, $0 < D < 360$). Воспользоваться следующим соотношением: $180^\circ = \pi$ радианов. В качестве значения π использовать 3.14. С помощью функции DegToRad перевести из градусов в радианы пять данных углов.

Proc33. Описать функцию RadToDeg(R) вещественного типа, находящую величину угла в градусах, если дана его величина R в радианах (R — вещественное число, $0 < R < 2\cdot\pi$). Воспользоваться следующим соотношением: $180^\circ = \pi$ радианов. В качестве значения π использовать 3.14. С помощью функции RadToDeg перевести из радианов в градусы пять данных углов.

Proc34. Описать функцию Fact(N) вещественного типа, вычисляющую значение *факториала* $N! = 1\cdot2\cdot\ldots\cdot N$ ($N > 0$ — параметр целого типа; вещественное возвращаемое значение используется для того, чтобы избежать целочисленного переполнения при больших значениях N). С помощью этой функции найти факториалы пяти данных целых чисел.

Proc35. Описать функцию Fact2(N) вещественного типа, вычисляющую *двойной факториал*:

$N!! = 1\cdot3\cdot5\cdot\ldots\cdot N$, если N — нечетное;

$N!! = 2\cdot4\cdot6\cdot\ldots\cdot N$, если N — четное

($N > 0$ — параметр целого типа; вещественное возвращаемое значение используется для того, чтобы избежать целочисленного переполнения при больших значениях N). С помощью этой функции найти двойные факториалы пяти данных целых чисел.

Proc36. Описать функцию Fib(N) целого типа, вычисляющую N-й элемент последовательности *чисел Фибоначчи* F_K, которая описывается следующими формулами:

$$F_1 = 1, \qquad F_2 = 1, \qquad F_K = F_{K-2} + F_{K-1}, \quad K = 3, 4, \ldots .$$

Используя функцию Fib, найти пять чисел Фибоначчи с данными номерами $N_1, N_2, \ldots, N_5$.

Дополнительные задания на процедуры и функции

Proc37. Описать функцию Power1(A, B) вещественного типа, находящую величину A^B по формуле $A^B = \exp(B\cdot\ln(A))$ (параметры A и B — вещественные).

В случае нулевого или отрицательного параметра A функция возвращает 0. С помощью этой функции найти степени A^P, B^P, C^P, если даны числа P, A, B, C.

Proc38. Описать функцию Power2(A, N) вещественного типа, находящую величину A^N (A — вещественный, N — целый параметр) по следующим формулам:

$A^0 = 1$;

$A^N = A \cdot A \cdot \ldots \cdot A$ (N сомножителей), если $N > 0$;

$A^N = 1/(A \cdot A \cdot \ldots \cdot A)$ ($|N|$ сомножителей), если $N < 0$.

С помощью этой функции найти A^K, A^L, A^M, если даны числа A, K, L, M.

Proc39. Используя функции Power1 и Power2 из Proc37 и Proc38, описать функцию Power3(A, B) вещественного типа с вещественными параметрами, находящую A^B следующим образом: если B имеет нулевую дробную часть, то вызывается функция Power2(A, Round(B)); в противном случае вызывается функция Power1(A, B). С помощью этой функции найти A^P, B^P, C^P, если даны числа P, A, B, C.

Proc40°. Описать функцию Exp1(x, ε) вещественного типа (параметры x, ε — вещественные, $\varepsilon > 0$), находящую приближенное значение функции $\exp(x)$:

$$\exp(x) = 1 + x + x^2/(2!) + x^3/(3!) + \ldots + x^n/(n!) + \ldots$$

($n! = 1 \cdot 2 \cdot \ldots \cdot n$). В сумме учитывать все слагаемые, большие ε. С помощью Exp1 найти приближенное значение экспоненты для данного x при шести данных ε.

Proc41. Описать функцию Sin1(x, ε) вещественного типа (параметры x, ε — вещественные, $\varepsilon > 0$), находящую приближенное значение функции $\sin(x)$:

$$\sin(x) = x - x^3/(3!) + x^5/(5!) - \ldots + (-1)^n \cdot x^{2 \cdot n+1}/((2 \cdot n+1)!) + \ldots .$$

В сумме учитывать все слагаемые, модуль которых больше ε. С помощью Sin1 найти приближенное значение синуса для данного x при шести данных ε.

Proc42. Описать функцию Cos1(x, ε) вещественного типа (параметры x, ε — вещественные, $\varepsilon > 0$), находящую приближенное значение функции $\cos(x)$:

$$\cos(x) = 1 - x^2/(2!) + x^4/(4!) - \ldots + (-1)^n \cdot x^{2 \cdot n}/((2 \cdot n)!) + \ldots .$$

В сумме учитывать все слагаемые, модуль которых больше ε. С помощью Cos1 найти приближенное значение косинуса для данного x при шести данных ε.

Proc43. Описать функцию Ln1(x, ε) вещественного типа (параметры x, ε —

вещественные, $|x|<1$, $\varepsilon>0$), находящую приближенное значение функции $\ln(1+x)$:

$$\ln(1+x) = x - x^2/2 + x^3/3 - \ldots + (-1)^n \cdot x^{n+1}/(n+1) + \ldots .$$

В сумме учитывать все слагаемые, модуль которых больше ε. С помощью Ln1 найти приближенное значение $\ln(1 + x)$ для данного x при шести данных ε.

Proc44. Описать функцию Arctg1(x, ε) вещественного типа (параметры x, ε — вещественные, $|x|<1$, $\varepsilon>0$), находящую приближенное значение функции $\operatorname{arctg}(x)$:

$$\operatorname{arctg}(x) = x - x^3/3 + x^5/5 - \ldots + (-1)^n \cdot x^{2\cdot n+1}/(2\cdot n+1) + \ldots .$$

В сумме учитывать все слагаемые, модуль которых больше ε. С помощью Arctg1 найти приближенное значение $\operatorname{arctg}(x)$ для данного x при шести данных ε.

Proc45. Описать функцию Power4(x, a, ε) вещественного типа (параметры x, a, ε — вещественные, $|x|<1; a, \varepsilon>0$), находящую приближенное значение функции $(1+x)^a$:

$$(1+x)^a = 1 + a\cdot x + a\cdot(a-1)\cdot x^2/(2!) + \ldots + a\cdot(a-1)\cdot\ldots\cdot(a-n+1)\cdot x^n/(n!) + \ldots .$$

В сумме учитывать все слагаемые, модуль которых больше ε. С помощью Power4 найти приближенное значение $(1+x)^a$ для данных x и a при шести данных ε.

Proc46. Описать функцию NOD2(A, B) целого типа, находящую *наибольший общий делитель* (НОД) двух целых положительных чисел A и B, используя *алгоритм Евклида*:

$$\text{НОД}(A, B) = \text{НОД}(B, A \bmod B), \quad \text{если } B \neq 0; \qquad \text{НОД}(A, 0) = A.$$

С помощью этой функции найти наибольшие общие делители пар (A, B), (A, C), (A, D), если даны числа A, B, C, D.

Proc47. Используя функцию NOD2 (см. Proc46), описать процедуру Frac1(a, b, p, q), преобразующую дробь a/b к несократимому виду p/q (все параметры процедуры — целого типа, a и b — входные, p и q — выходные). Знак результирующей дроби p/q приписывается числителю (т. е. $q > 0$). С помощью Frac1 найти несократимые дроби, равные $a/b + c/d$, $a/b + e/f$, $a/b + g/h$ (числа a, b, c, d, e, f, g, h даны).

Proc48. Учитывая, что *наименьшее общее кратное* двух целых положительных чисел A и B равно $A\cdot(B/\text{НОД}(A, B))$, где НОД($A$, B) — наибольший общий делитель A и B, и используя функцию NOD2 (см. Proc46), описать функцию NOK2(A, B) целого типа, находящую наименьшее общее крат-

ное чисел A и B. С помощью NOK2 найти наименьшие общие кратные пар (A, B), (A, C), (A, D), если даны числа A, B, C, D.

Proc49. Учитывая соотношение НОД(A, B, C) = НОД(НОД(A, B), C) и используя функцию NOD2 (см. Proc46), описать функцию NOD3(A, B, C) целого типа, находящую наибольший общий делитель трех целых положительных чисел A, B, C. С помощью этой функции найти наибольшие общие делители троек (A, B, C), (A, C, D) и (B, C, D), если даны числа A, B, C, D.

Proc50. Описать процедуру TimeToHMS(T, H, M, S), определяющую по времени T (в секундах) содержащееся в нем количество часов H, минут M и секунд S (T — входной, H, M и S — выходные параметры целого типа). Используя эту процедуру, найти количество часов, минут и секунд для пяти данных отрезков времени T_1, T_2, ..., T_5.

Proc51. Описать процедуру IncTime(H, M, S, T), которая увеличивает на T секунд время, заданное в часах H, минутах M и секундах S (H, M и S — входные и выходные параметры, T — входной параметр; все параметры — целые положительные). Дано время (в часах H, минутах M, секундах S) и целое число T. Используя процедуру IncTime, увеличить данное время на T секунд и вывести новые значения H, M, S.

Proc52. Описать функцию IsLeapYear(Y) логического типа, которая возвращает TRUE, если год Y (целое положительное число) является високосным, и FALSE в противном случае. Вывести значение функции IsLeapYear для пяти данных значений параметра Y. *Високосным* считается год, делящийся на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400.

Proc53. Используя функцию IsLeapYear из задания Proc52, описать функцию MonthDays(M, Y) целого типа, которая возвращает количество дней для M-го месяца года Y ($1 \le M \le 12$, $Y > 0$ — целые числа). Вывести значение функции MonthDays для данного года Y и месяцев M_1, M_2, M_3.

Proc54. Используя функцию MonthDays из задания Proc53, описать процедуру PrevDate(D, M, Y), которая по информации о правильной дате, включающей день D, номер месяца M и год Y, определяет предыдущую дату (параметры целого типа D, M, Y являются одновременно входными и выходными). Применить процедуру PrevDate к трем исходным датам и вывести полученные значения предыдущих дат.

Proc55. Используя функцию MonthDays из задания Proc53, описать процедуру NextDate(D, M, Y), которая по информации о правильной дате, включа-

ющей день D, номер месяца M и год Y, определяет следующую дату (параметры целого типа D, M, Y являются одновременно входными и выходными). Применить процедуру NextDate к трем исходным датам и вывести полученные значения следующих дат.

Proc56. Описать функцию Leng(x_A, y_A, x_B, y_B) вещественного типа, находящую длину отрезка AB на плоскости по координатам его концов:

$$|AB| = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}$$

(x_A, y_A, x_B, y_B — вещественные параметры). С помощью этой функции найти длины отрезков AB, AC, AD, если даны координаты точек A, B, C, D.

Proc57. Используя функцию Leng из задания Proc56, описать функцию Perim(x_A, y_A, x_B, y_B, x_C, y_C) вещественного типа, находящую периметр треугольника ABC по координатам его вершин (x_A, y_A, x_B, y_B, x_C, y_C — вещественные параметры). С помощью этой функции найти периметры треугольников ABC, ABD, ACD, если даны координаты точек A, B, C, D.

Proc58. Используя функции Leng и Perim из заданий Proc56 и Proc57, описать функцию Area(x_A, y_A, x_B, y_B, x_C, y_C) вещественного типа, находящую площадь треугольника ABC по формуле

$$S_{ABC} = \sqrt{p\cdot(p-|AB|)\cdot(p-|AC|)\cdot(p-|BC|)},$$

где p — *полупериметр*. С помощью этой функции найти площади треугольников ABC, ABD, ACD, если даны координаты точек A, B, C, D.

Proc59. Используя функции Leng и Area из заданий Proc56 и Proc58, описать функцию Dist(x_P, y_P, x_A, y_A, x_B, y_B) вещественного типа, находящую расстояние $D(P, AB)$ от точки P до прямой AB по формуле

$$D(P, AB) = 2\cdot S_{PAB}/|AB|,$$

где S_{PAB} — площадь треугольника PAB. С помощью этой функции найти расстояния от точки P до прямых AB, AC, BC, если даны координаты точек P, A, B, C.

Proc60. Используя функцию Dist из задания Proc59, описать процедуру Heights($x_A, y_A, x_B, y_B, x_C, y_C, h_A, h_B, h_C$), находящую высоты h_A, h_B, h_C треугольника ABC (выходные параметры), проведенные соответственно из вершин A, B, C (их координаты являются входными параметрами). С помощью этой процедуры найти высоты треугольников ABC, ABD, ACD, если даны координаты точек A, B, C, D.

Минимумы и максимумы

Для решения заданий из данной группы следует использовать «однопроходные» алгоритмы, позволяющие получить требуемый результат после *однократного* просмотра набора исходных данных. Однопроходные алгоритмы обладают важным преимуществом: для них не требуется хранить в памяти одновременно весь набор данных, поэтому при программной реализации этих алгоритмов *можно не использовать массивы*.

Во всех заданиях данной группы предполагается, что исходный набор содержит ненулевое количество элементов (в частности, число N всегда больше нуля).

Minmax1°. Дано целое число N и набор из N чисел. Найти минимальный и максимальный из элементов данного набора и вывести их в указанном порядке.

Minmax2. Дано целое число N и набор из N прямоугольников, заданных своими сторонами — парами чисел (a, b). Найти минимальную площадь прямоугольника из данного набора.

Minmax3. Дано целое число N и набор из N прямоугольников, заданных своими сторонами — парами чисел (a, b). Найти максимальный периметр прямоугольника из данного набора.

Minmax4°. Дано целое число N и набор из N чисел. Найти номер минимального элемента из данного набора.

Minmax5. Дано целое число N и набор из N пар чисел (m, v) — данные о массе m и объеме v деталей, изготовленных из различных материалов. Вывести номер детали, изготовленной из материала максимальной плотности, а также величину этой максимальной плотности. Плотность P вычисляется по формуле

$$P = m/v.$$

Minmax6°. Дано целое число N и набор из N целых чисел. Найти номера первого минимального и последнего максимального элемента из данного набора и вывести их в указанном порядке.

Minmax7. Дано целое число N и набор из N целых чисел. Найти номера первого максимального и последнего минимального элемента из данного набора и вывести их в указанном порядке.

Minmax8. Дано целое число N и набор из N целых чисел. Найти номера первого и последнего минимального элемента из данного набора и вывести

их в указанном порядке.

Minmax9. Дано целое число N и набор из N целых чисел. Найти номера первого и последнего максимального элемента из данного набора и вывести их в указанном порядке.

Minmax10. Дано целое число N и набор из N целых чисел. Найти номер первого *экстремального* (то есть минимального или максимального) элемента из данного набора.

Minmax11. Дано целое число N и набор из N целых чисел. Найти номер последнего *экстремального* (то есть минимального или максимального) элемента из данного набора.

Minmax12°. Дано целое число N и набор из N чисел. Найти минимальное положительное число из данного набора. Если положительные числа в наборе отсутствуют, то вывести 0.

Minmax13. Дано целое число N и набор из N целых чисел. Найти номер первого максимального нечетного числа из данного набора. Если нечетные числа в наборе отсутствуют, то вывести 0.

Minmax14. Дано число B (>0) и набор из десяти чисел. Вывести минимальный из тех элементов набора, которые больше B, а также его номер. Если чисел, больших B, в наборе нет, то дважды вывести 0

Minmax15. Даны числа B, C ($0 < B < C$) и набор из десяти чисел. Вывести максимальный из элементов набора, содержащихся в интервале (B, C), и его номер. Если требуемые числа в наборе отсутствуют, то дважды вывести 0.

Minmax16. Дано целое число N и набор из N целых чисел. Найти количество элементов, расположенных перед первым минимальным элементом.

Minmax17. Дано целое число N и набор из N целых чисел. Найти количество элементов, расположенных после последнего максимального элемента.

Minmax18. Дано целое число N и набор из N целых чисел. Найти количество элементов, содержащихся между первым и последним максимальным элементом. Если в наборе имеется единственный максимальный элемент, то вывести 0.

Minmax19°. Дано целое число N и набор из N целых чисел. Найти количество минимальных элементов из данного набора.

Minmax20. Дано целое число N и набор из N целых чисел. Найти общее количество *экстремальных* (то есть минимальных и максимальных) элементов из данного набора.

Minmax21. Дано целое число N (>2) и набор из N чисел — значений некоторой величины, полученных в N опытах. Найти среднее значение этой величины. При вычислении среднего значения не учитывать минимальное и максимальное из имеющихся в наборе значений.

Minmax22°. Дано целое число N (>2) и набор из N чисел. Найти два наименьших элемента из данного набора и вывести эти элементы в порядке возрастания их значений.

Minmax23. Дано целое число N (>3) и набор из N чисел. Найти три наибольших элемента из данного набора и вывести эти элементы в порядке убывания их значений.

Minmax24. Дано целое число N (>1) и набор из N чисел. Найти максимальную сумму двух соседних чисел из данного набора.

Minmax25. Дано целое число N (>1) и набор из N чисел. Найти номера двух соседних чисел из данного набора, произведение которых является минимальным, и вывести вначале меньший, а затем больший номер.

Minmax26. Дано целое число N и набор из N целых чисел. Найти максимальное количество четных чисел в наборе, идущих подряд. Если четные числа в наборе отсутствуют, то вывести 0.

Minmax27. Дано целое число N и набор из N целых чисел, содержащий только нули и единицы. Найти номер элемента, с которого начинается самая длинная последовательность одинаковых чисел, и количество элементов в этой последовательности. Если таких последовательностей несколько, то вывести номер первой из них.

Minmax28. Дано целое число N и набор из N целых чисел, содержащий только нули и единицы. Найти номер элемента, с которого начинается самая длинная последовательность единиц, и количество элементов в этой последовательности. Если таких последовательностей несколько, то вывести номер последней из них. Если единицы в исходном наборе отсутствуют, то дважды вывести 0.

Minmax29. Дано целое число N и набор из N целых чисел. Найти максимальное количество подряд идущих минимальных элементов из данного набора.

Minmax30. Дано целое число N и набор из N целых чисел. Найти минимальное количество подряд идущих максимальных элементов из данного набора.

Одномерные массивы

Условие вида «дан массив размера N» означает, что вначале дается *фактический размер* массива (целое число N), а затем приводятся все его элементы. Если в задании явно не указывается, какие значения может принимать размер исходного массива, то предполагается, что размер может изменяться в пределах от 2 до 10. Индекс начального элемента массива считается равным 1.

Если в задании, связанном с созданием (преобразованием) массива, не описан результирующий набор данных, то предполагается, что этим набором является созданный (преобразованный) массив, и необходимо вывести все его элементы в порядке возрастания их индексов.

Формирование массива и вывод его элементов

В заданиях на формирование массива предполагается, что размер результирующего массива не превосходит 10.

Array1. Дано целое число N (>0). Сформировать и вывести целочисленный массив размера N, содержащий N первых положительных нечетных чисел: 1, 3, 5, ...

Array2. Дано целое число N (>0). Сформировать и вывести целочисленный массив размера N, содержащий степени двойки от первой до N-й: 2, 4, 8, 16, ...

Array3. Дано целое число N (>1), а также первый член A и разность D *арифметической прогрессии*. Сформировать и вывести массив размера N, содержащий N первых членов данной прогрессии:

$$A, \quad A + D, \quad A + 2 \cdot D, \quad A + 3 \cdot D, \quad \ldots .$$

Array4°. Дано целое число N (>1), а также первый член A и знаменатель D *геометрической прогрессии*. Сформировать и вывести массив размера N, содержащий N первых членов данной прогрессии:

$$A, \quad A \cdot D, \quad A \cdot D^2, \quad A \cdot D^3, \quad \ldots .$$

Array5. Дано целое число N (>2). Сформировать и вывести целочисленный массив размера N, содержащий N первых элементов последовательности *чисел Фибоначчи* F_K:

$$F_1 = 1, \qquad F_2 = 1, \qquad F_K = F_{K-2} + F_{K-1}, \quad K = 3, 4, \ldots .$$

Array6. Даны целые числа N (>2), A и B. Сформировать и вывести целочисленный массив размера N, первый элемент которого равен A, второй

равен B, а каждый последующий элемент равен сумме всех предыдущих.

Array7°. Дан массив размера N. Вывести его элементы в обратном порядке.

Array8. Дан целочисленный массив размера N. Вывести все содержащиеся в данном массиве нечетные числа в порядке возрастания их индексов, а также их количество K.

Array9. Дан целочисленный массив размера N. Вывести все содержащиеся в данном массиве четные числа в порядке убывания их индексов, а также их количество K.

Array10. Дан целочисленный массив размера N. Вывести вначале все содержащиеся в данном массиве четные числа в порядке возрастания их индексов, а затем — все нечетные числа в порядке убывания их индексов.

Array11. Дан массив A размера N и целое число K ($1 \le K \le N$). Вывести элементы массива с порядковыми номерами, кратными K: $A_K, A_{2\cdot K}, A_{3\cdot K}, \ldots$. Условный оператор не использовать.

Array12. Дан массив A размера N (N — четное число). Вывести его элементы с четными номерами в порядке возрастания номеров: A_2, A_4, A_6, …, A_N. Условный оператор не использовать.

Array13. Дан массив A размера N (N — нечетное число). Вывести его элементы с нечетными номерами в порядке убывания номеров: A_N, A_{N-2}, A_{N-4}, …, A_1. Условный оператор не использовать.

Array14. Дан массив A размера N. Вывести вначале его элементы с четными номерами (в порядке возрастания номеров), а затем — элементы с нечетными номерами (также в порядке возрастания номеров):

$$A_2, \quad A_4, \quad A_6, \quad \ldots, \quad A_1, \quad A_3, \quad A_5, \quad \ldots.$$

Условный оператор не использовать.

Array15. Дан массив A размера N. Вывести вначале его элементы с нечетными номерами в порядке возрастания номеров, а затем — элементы с четными номерами в порядке убывания номеров:

$$A_1, \quad A_3, \quad A_5, \quad \ldots, \quad A_6, \quad A_4, \quad A_2.$$

Условный оператор не использовать.

Array16°. Дан массив A размера N. Вывести его элементы в следующем порядке:

$$A_1, \quad A_N, \quad A_2, \quad A_{N-1}, \quad A_3, \quad A_{N-2}, \quad \ldots.$$

Array17. Дан массив A размера N. Вывести его элементы в следующем порядке:

$$A_1, \quad A_2, \quad A_N, \quad A_{N-1}, \quad A_3, \quad A_4, \quad A_{N-2}, \quad A_{N-3}, \quad \ldots.$$

Анализ элементов массива

Для выполнения некоторых заданий из данного пункта не требуется одновременно хранить в памяти все исходные данные, поэтому использовать при их выполнении массивы, строго говоря, *не нужно*. Однако применение массивов позволяет сделать алгоритмы решения этих заданий более простыми и наглядными. Задания из данного пункта можно дополнить заданиями из групп Series и Minmax, рассматривая их как задания на обработку массивов. С другой стороны, для тех заданий данного пункта, которые можно выполнить, не используя массивы, полезно реализовать и такие алгоритмы решения.

Array18. Дан массив A ненулевых целых чисел размера 10. Вывести значение первого из тех его элементов A_K, которые удовлетворяют неравенству $A_K < A_{10}$. Если таких элементов нет, то вывести 0.

Array19. Дан целочисленный массив A размера 10. Вывести порядковый номер последнего из тех его элементов A_K, которые удовлетворяют двойному неравенству $A_1 < A_K < A_{10}$. Если таких элементов нет, то вывести 0.

Array20. Дан массив размера N и целые числа K и L ($1 \le K \le L \le N$). Найти сумму элементов массива с номерами от K до L включительно.

Array21. Дан массив размера N и целые числа K и L ($1 \le K \le L \le N$). Найти среднее арифметическое элементов массива с номерами от K до L включительно.

Array22. Дан массив размера N и целые числа K и L ($1 < K \le L \le N$). Найти сумму всех элементов массива, кроме элементов с номерами от K до L включительно.

Array23. Дан массив размера N и целые числа K и L ($1 < K \le L \le N$). Найти среднее арифметическое всех элементов массива, кроме элементов с номерами от K до L включительно.

Array24. Дан целочисленный массив размера N, не содержащий одинаковых чисел. Проверить, образуют ли его элементы *арифметическую прогрессию* (см. задание Array3). Если образуют, то вывести разность прогрессии, если нет — вывести 0.

Array25. Дан массив ненулевых целых чисел размера N. Проверить, образуют ли его элементы *геометрическую прогрессию* (см. задание Array4). Если образуют, то вывести знаменатель прогрессии, если нет — вывести 0.

Array26. Дан целочисленный массив размера N. Проверить, чередуются ли в нем четные и нечетные числа. Если чередуются, то вывести 0, если нет,

то вывести порядковый номер первого элемента, нарушающего закономерность.

Array27. Дан массив ненулевых целых чисел размера N. Проверить, чередуются ли в нем положительные и отрицательные числа. Если чередуются, то вывести 0, если нет, то вывести порядковый номер первого элемента, нарушающего закономерность.

Array28. Дан массив A размера N. Найти минимальный элемент из его элементов с четными номерами: A_2, A_4, A_6, ...

Array29. Дан массив A размера N. Найти максимальный элемент из его элементов с нечетными номерами: A_1, A_3, A_5, ...

Array30. Дан массив размера N. Найти номера тех элементов массива, которые больше своего правого соседа, и количество таких элементов. Найденные номера выводить в порядке их возрастания.

Array31. Дан массив размера N. Найти номера тех элементов массива, которые больше своего левого соседа, и количество таких элементов. Найденные номера выводить в порядке их убывания.

Array32°. Дан массив размера N. Найти номер его первого локального минимума (*локальный минимум* — это элемент, который меньше любого из своих соседей).

Array33. Дан массив размера N. Найти номер его последнего локального максимума (*локальный максимум* — это элемент, который больше любого из своих соседей).

Array34. Дан массив размера N. Найти максимальный из его локальных минимумов (определение *локального минимума* дано в задании Array32).

Array35. Дан массив размера N. Найти минимальный из его локальных максимумов (определение *локального максимума* дано в задании Array33).

Array36. Дан массив размера N. Найти максимальный из его элементов, не являющихся ни локальным минимумом, ни локальным максимумом (определения *локального минимума* и *локального максимума* даны в заданиях Array32 и Array33). Если таких элементов в массиве нет, то вывести 0 (как вещественное число).

Array37. Дан массив размера N. Найти количество участков, на которых его элементы возрастают.

Array38. Дан массив размера N. Найти количество участков, на которых его элементы убывают.

Array39. Дан массив размера N. Найти количество его *промежутков моно-*

тонности (то есть участков, на которых его элементы возрастают или убывают).

Array40. Дано число R и массив A размера N. Найти элемент массива, который *наиболее близок* к числу R (то есть такой элемент A_K, для которого величина $|A_K - R|$ является минимальной).

Array41. Дан массив размера N. Найти два соседних элемента, сумма которых максимальна, и вывести эти элементы в порядке возрастания их индексов.

Array42. Дано число R и массив размера N. Найти два соседних элемента массива, сумма которых наиболее близка к числу R, и вывести эти элементы в порядке возрастания их индексов (определение наиболее близких чисел дано в задании Array40).

Array43. Дан целочисленный массив размера N, все элементы которого упорядочены (по возрастанию или по убыванию). Найти количество различных элементов в данном массиве.

Array44. Дан целочисленный массив размера N, содержащий ровно два одинаковых элемента. Найти номера одинаковых элементов и вывести эти номера в порядке возрастания.

Array45. Дан массив размера N. Найти номера двух ближайших элементов из этого массива (то есть элементов с наименьшим модулем разности) и вывести эти номера в порядке возрастания.

Array46. Дано число R и массив размера N. Найти два различных элемента массива, сумма которых наиболее близка к числу R, и вывести эти элементы в порядке возрастания их индексов (определение наиболее близких чисел дано в задании Array40).

Array47°. Дан целочисленный массив размера N. Найти количество различных элементов в данном массиве.

Array48. Дан целочисленный массив размера N. Найти максимальное количество его одинаковых элементов.

Array49. Дан целочисленный массив размера N. Если он является *перестановкой*, то есть содержит все числа от 1 до N, то вывести 0; в противном случае вывести номер первого недопустимого элемента.

Array50. Дан целочисленный массив A размера N, являющийся перестановкой (определение *перестановки* дано в задании Array49). Найти количество *инверсий* в данной перестановке, то есть таких пар элементов A_I и A_J, в которых большее число находится слева от меньшего: $A_I > A_J$ при $I < J$.

Работа с несколькими массивами

Array51. Даны массивы A и B одинакового размера N. Поменять местами их содержимое и вывести вначале элементы преобразованного массива A, а затем — элементы преобразованного массива B.

Array52. Дан массив A размера N. Сформировать новый массив B того же размера, элементы которого определяются следующим образом:

$$B_K = \begin{cases} 2 \cdot A_K, & \text{если } A_K < 5, \\ A_K/2 & \text{в противном случае.} \end{cases}$$

Array53. Даны два массива A и B одинакового размера N. Сформировать новый массив C того же размера, каждый элемент которого равен максимальному из элементов массивов A и B с тем же индексом.

Array54°. Дан целочисленный массив A размера N. Переписать в новый целочисленный массив B все четные числа из исходного массива (в том же порядке) и вывести размер полученного массива B и его содержимое.

Array55. Дан целочисленный массив A размера N (≤ 15). Переписать в новый целочисленный массив B все элементы с нечетными порядковыми номерами (1, 3, . . .) и вывести размер полученного массива B и его содержимое. Условный оператор не использовать.

Array56. Дан целочисленный массив A размера N (≤ 15). Переписать в новый целочисленный массив B все элементы с порядковыми номерами, кратными трем (3, 6, . . .), и вывести размер полученного массива B и его содержимое. Условный оператор не использовать.

Array57. Дан целочисленный массив A размера N. Переписать в новый целочисленный массив B того же размера вначале все элементы исходного массива с четными номерами, а затем — с нечетными:

$$A_2, \quad A_4, \quad A_6, \quad \ldots, \quad A_1, \quad A_3, \quad A_5, \quad \ldots .$$

Условный оператор не использовать.

Array58. Дан массив A размера N. Сформировать новый массив B того же размера по следующему правилу: элемент B_K равен сумме элементов массива A с номерами от 1 до K.

Array59. Дан массив A размера N. Сформировать новый массив B того же размера по следующему правилу: элемент B_K равен среднему арифметическому элементов массива A с номерами от 1 до K.

Array60. Дан массив A размера N. Сформировать новый массив B того же размера по следующему правилу: элемент B_K равен сумме элементов

массива A с номерами от K до N.

Array61. Дан массив A размера N. Сформировать новый массив B того же размера по следующему правилу: элемент B_K равен среднему арифметическому элементов массива A с номерами от K до N.

Array62. Дан массив A размера N. Сформировать два новых массива B и C: в массив B записать все положительные элементы массива A, в массив C — все отрицательные (сохраняя исходный порядок следования элементов). Вывести вначале размер и содержимое массива B, а затем — размер и содержимое массива C.

Array63°. Даны два массива A и B размера 5, элементы которых упорядочены по возрастанию. Объединить эти массивы так, чтобы результирующий массив C (размера 10) остался упорядоченным по возрастанию.

Array64. Даны три целочисленных массива A, B и C размера N_A, N_B, N_C соответственно, элементы которых упорядочены по убыванию. Объединить эти массивы так, чтобы результирующий целочисленный массив D (размера $N_A + N_B + N_C$) остался упорядоченным по убыванию.

Преобразование массива

При выполнении заданий из данного пункта не следует использовать вспомогательные массивы.

Изменение элементов массива

Array65. Дан массив A размера N и целое число K ($1 \le K \le N$). Преобразовать массив, увеличив каждый его элемент на исходное значение элемента A_K.

Array66. Дан целочисленный массив размера N. Увеличить все четные числа, содержащиеся в массиве, на исходное значение первого четного числа. Если четные числа в массиве отсутствуют, то оставить массив без изменений.

Array67. Дан целочисленный массив размера N. Увеличить все нечетные числа, содержащиеся в массиве, на исходное значение последнего нечетного числа. Если нечетные числа в массиве отсутствуют, то оставить массив без изменений.

Array68. Дан массив размера N. Поменять местами его минимальный и максимальный элементы.

Array69. Дан массив размера N (N — четное число). Поменять местами его первый элемент со вторым, третий — с четвертым и т. д.

Array70. Дан массив размера N (N — четное число). Поменять местами первую и вторую половины массива.

Array71°. Дан массив размера N. Поменять порядок его элементов на обратный.

Array72. Дан массив A размера N и целые числа K и L ($1 \le K < L \le N$). Переставить в обратном порядке элементы массива, расположенные между элементами A_K и A_L, включая эти элементы.

Array73. Дан массив A размера N и целые числа K и L ($1 \le K < L \le N$). Переставить в обратном порядке элементы массива, расположенные между элементами A_K и A_L, не включая эти элементы.

Array74. Дан массив размера N. Обнулить элементы массива, расположенные между его минимальным и максимальным элементами (не включая минимальный и максимальный элементы).

Array75. Дан массив размера N. Переставить в обратном порядке элементы массива, расположенные между его минимальным и максимальным элементами, включая минимальный и максимальный элементы.

Array76. Дан массив размера N. Обнулить все его *локальные максимумы* (то есть числа, большие своих соседей).

Array77. Дан массив размера N. Возвести в квадрат все его *локальные минимумы* (то есть числа, меньшие своих соседей).

Array78. Дан массив размера N. Заменить каждый элемент массива на среднее арифметическое этого элемента и его соседей.

Array79°. Дан массив размера N. Осуществить *сдвиг* элементов массива вправо на одну позицию (при этом A_1 перейдет в A_2, A_2 — в A_3, …, A_{N-1} — в A_N, а исходное значение последнего элемента будет потеряно). Первый элемент полученного массива положить равным 0.

Array80. Дан массив размера N. Осуществить *сдвиг* элементов массива влево на одну позицию (при этом A_N перейдет в A_{N-1}, A_{N-1} — в A_{N-2}, …, A_2 — в A_1, а исходное значение первого элемента будет потеряно). Последний элемент полученного массива положить равным 0.

Array81. Дан массив размера N и целое число K ($1 \le K < N$). Осуществить *сдвиг* элементов массива вправо на K позиций (при этом A_1 перейдет в A_{K+1}, A_2 — в A_{K+2}, …, A_{N-K} — в A_N, а исходное значение K последних элементов будет потеряно). Первые K элементов полученного массива

положить равными 0.

Array82. Дан массив размера N и целое число K ($1 \le K < N$). Осуществить *сдвиг* элементов массива влево на K позиций (при этом A_N перейдет в A_{N-K}, A_{N-1} — в A_{N-K-1}, ..., A_{K+1} — в A_1, а исходное значение K первых элементов будет потеряно). Последние K элементов полученного массива положить равными 0.

Array83. Дан массив размера N. Осуществить *циклический сдвиг* элементов массива вправо на одну позицию (при этом A_1 перейдет в A_2, A_2 — в A_3, ..., A_N — в A_1).

Array84. Дан массив размера N. Осуществить *циклический сдвиг* элементов массива влево на одну позицию (при этом A_N перейдет в A_{N-1}, A_{N-1} — в A_{N-2}, ..., A_1 — в A_N).

Array85. Дан массив A размера N и целое число K ($1 \le K \le 4$, $K < N$). Осуществить *циклический сдвиг* элементов массива вправо на K позиций (при этом A_1 перейдет в A_{K+1}, A_2 — в A_{K+2}, ..., A_N — в A_K). Допускается использовать вспомогательный массив из 4 элементов.

Array86. Дан массив A размера N и целое число K ($1 \le K \le 4$, $K < N$). Осуществить *циклический сдвиг* элементов массива влево на K позиций (при этом A_N перейдет в A_{N-K}, A_{N-1} — в A_{N-K-1}, ..., A_1 — в A_{N-K+1}). Допускается использовать вспомогательный массив из 4 элементов.

Array87. Дан массив размера N, все элементы которого, кроме первого, упорядочены по возрастанию. Сделать массив упорядоченным, переместив первый элемент на новую позицию.

Array88. Дан массив размера N, все элементы которого, кроме последнего, упорядочены по возрастанию. Сделать массив упорядоченным, переместив последний элемент на новую позицию.

Array89°. Дан массив размера N, все элементы которого, кроме одного, упорядочены по убыванию. Сделать массив упорядоченным, переместив элемент, нарушающий упорядоченность, на новую позицию.

Удаление и вставка элементов

Array90. Дан массив размера N и целое число K ($1 \le K \le N$). Удалить из массива элемент с порядковым номером K.

Array91. Дан массив размера N и целые числа K и L ($1 \le K < L \le N$). Удалить из массива элементы с номерами от K до L включительно и вывести размер полученного массива и его содержимое.

Array92°. Дан целочисленный массив размера N. Удалить из массива все нечетные числа и вывести размер полученного массива и его содержимое.

Array93. Дан целочисленный массив размера N (>2). Удалить из массива все элементы с четными номерами (2, 4, ...). Условный оператор не использовать.

Array94. Дан целочисленный массив размера N (>2). Удалить из массива все элементы с нечетными номерами (1, 3, ...). Условный оператор не использовать.

Array95. Дан целочисленный массив размера N. Удалить из массива все соседние одинаковые элементы, оставив их первые вхождения.

Array96. Дан целочисленный массив размера N. Удалить из массива все одинаковые элементы, оставив их первые вхождения.

Array97. Дан целочисленный массив размера N. Удалить из массива все одинаковые элементы, оставив их последние вхождения.

Array98. Дан целочисленный массив размера N. Удалить из массива все элементы, встречающиеся менее трех раз, и вывести размер полученного массива и его содержимое.

Array99. Дан целочисленный массив размера N. Удалить из массива все элементы, встречающиеся более двух раз, и вывести размер полученного массива и его содержимое.

Array100. Дан целочисленный массив размера N. Удалить из массива все элементы, встречающиеся ровно два раза, и вывести размер полученного массива и его содержимое.

Array101. Дан массив размера N и целое число K ($1 \le K \le N$). Перед элементом массива с порядковым номером K вставить новый элемент с нулевым значением.

Array102. Дан массив размера N и целое число K ($1 \le K \le N$). После элемента массива с порядковым номером K вставить новый элемент с нулевым значением.

Array103. Дан массив размера N. Вставить элемент с нулевым значением перед минимальным и после максимального элемента массива.

Array104. Дан массив размера N и два целых числа K и M ($1 \le K \le N$, $1 \le M \le 10$). Перед элементом массива с номером K вставить M новых элементов с нулевыми значениями.

Array105. Дан массив размера N и два целых числа K и M ($1 \le K \le N$, $1 \le M \le 10$). После элемента массива с номером K вставить M новых

элементов с нулевыми значениями.

Array106. Дан массив размера N. Продублировать в нем элементы с четными номерами (2, 4, . . .). Условный оператор не использовать.

Array107. Дан массив размера N. Утроить в нем вхождения всех элементов с нечетными номерами (1, 3, . . .). Условный оператор не использовать.

Array108°. Дан массив размера N. Перед каждым положительным элементом массива вставить элемент с нулевым значением.

Array109. Дан массив размера N. После каждого отрицательного элемента массива вставить элемент с нулевым значением.

Array110. Дан целочисленный массив размера N. Продублировать в нем все четные числа.

Array111. Дан целочисленный массив размера N. Утроить в нем вхождения всех нечетных чисел.

Сортировка массива

Array112°. Дан массив A размера N (≤ 6). Упорядочить его по возрастанию методом сортировки *простым обменом* (*«пузырьковой» сортировкой*): просматривать массив, сравнивая его соседние элементы (A_1 и A_2, A_2 и A_3 и т. д.) и меняя их местами, если левый элемент пары больше правого; повторить описанные действия $N-1$ раз. Для контроля за выполняемыми действиями выводить содержимое массива после каждого просмотра. Учесть, что при каждом просмотре количество анализируемых пар можно уменьшить на 1.

Array113. Дан массив A размера N (≤ 6). Упорядочить его по возрастанию методом сортировки *простым выбором*: найти максимальный элемент массива и поменять его местами с последним (N-м) элементом; выполнить описанные действия $N-1$ раз, каждый раз уменьшая на 1 количество анализируемых элементов и выводя содержимое массива.

Array114. Дан массив A размера N (≤ 6). Упорядочить его по возрастанию методом сортировки *простыми вставками*: сравнить элементы A_1 и A_2 и, при необходимости меняя их местами, добиться того, чтобы они оказались упорядоченными по возрастанию; затем обратиться к элементу A_3 и переместить его в левую (уже упорядоченную) часть массива, сохранив ее упорядоченность; повторить этот процесс для остальных элементов, выводя содержимое массива после обработки каждого элемента (от 2-го до N-го). При выполнении описанных действий удобно использовать

прием «барьера», записывая очередной элемент перед его обработкой в дополнительный элемент массива A_0.

Array115. Дан массив A размера N. Не изменяя данный массив, вывести номера его элементов в том порядке, в котором соответствующие им элементы образуют возрастающую последовательность. Использовать метод *«пузырьковой» сортировки* (см. задание Array112), модифицировав его следующим образом: создать вспомогательный целочисленный *массив номеров I*, заполнив его числами от 1 до N; просматривать массив A, сравнивая пары элементов массива A с номерами I_1 и I_2, I_2 и I_3, ... и меняя местами соответствующие элементы массива I, если левый элемент пары больше правого. Повторив описанную процедуру просмотра $N-1$ раз, получим в массиве I требуемую последовательность номеров.

Серии целых чисел

Array116°. Дан целочисленный массив A размера N. Назовем *серией* группу подряд идущих одинаковых элементов, а *длиной серии* — количество этих элементов (длина серии может быть равна 1). Сформировать два новых целочисленных массива B и C одинакового размера, записав в массив B длины всех серий исходного массива, а в массив C — значения элементов, образующих эти серии.

Array117. Дан целочисленный массив размера N. Вставить перед каждой его серией элемент с нулевым значением (определение серии дано в задании Array116).

Array118. Дан целочисленный массив размера N. Вставить после каждой его серии элемент с нулевым значением (определение серии дано в задании Array116).

Array119. Дан целочисленный массив размера N. Преобразовать массив, увеличив каждую его серию на один элемент (определение серии дано в задании Array116).

Array120. Дан целочисленный массив размера N, содержащий по крайней мере одну серию, длина которой больше 1. Преобразовать массив, уменьшив каждую его серию на один элемент (определение серии дано в задании Array116).

Array121. Дано целое число K (>0) и целочисленный массив размера N. Преобразовать массив, удвоив длину его серии с номером K (определение серии дано в задании Array116). Если серий в массиве меньше K, то

вывести массив без изменений.

Array122. Дано целое число K (>1) и целочисленный массив размера N. Удалить из массива серию с номером K (определение серии дано в задании Array116). Если серий в массиве меньше K, то вывести массив без изменений.

Array123. Дано целое число K (>1) и целочисленный массив размера N. Поменять местами первую серию массива и его серию с номером K (определение серии дано в задании Array116). Если серий в массиве меньше K, то вывести массив без изменений.

Array124. Дано целое число K (>0) и целочисленный массив размера N. Поменять местами последнюю серию массива и его серию с номером K (определение серии дано в задании Array116). Если серий в массиве меньше K, то вывести массив без изменений.

Array125. Дано целое число L (>1) и целочисленный массив размера N. Заменить каждую серию массива, длина которой меньше L, на один элемент с нулевым значением (определение серии дано в задании Array116).

Array126. Дано целое число L (>0) и целочисленный массив размера N. Заменить каждую серию массива, длина которой равна L, на один элемент с нулевым значением (определение серии дано в задании Array116).

Array127. Дано целое число L (>0) и целочисленный массив размера N. Заменить каждую серию массива, длина которой больше L, на один элемент с нулевым значением (определение серии дано в задании Array116).

Array128. Дан целочисленный массив размера N. Преобразовать массив, увеличив его первую серию наибольшей длины на один элемент (определение серии дано в задании Array116).

Array129. Дан целочисленный массив размера N. Преобразовать массив, увеличив его последнюю серию наибольшей длины на один элемент (определение серии дано в задании Array116).

Array130. Дан целочисленный массив размера N. Преобразовать массив, увеличив все его серии наибольшей длины на один элемент (определение серии дано в задании Array116).

Множества точек на плоскости

Для хранения данных о каждом наборе точек следует использовать по два массива: первый массив для хранения абсцисс, второй — для хранения ординат. Можно также использовать массив *записей* с двумя полями (см. задание

Param64).

Array131. Дано множество A из N точек на плоскости и точка B (точки заданы своими координатами x, y). Найти точку из множества A, наиболее близкую к точке B. *Расстояние* R между точками с координатами (x_1, y_1) и (x_2, y_2) вычисляется по формуле:

$$R = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

Array132. Дано множество A из N точек (точки заданы своими координатами x, y). Среди всех точек этого множества, лежащих во второй четверти, найти точку, наиболее удаленную от начала координат. Если таких точек нет, то вывести точку с нулевыми координатами.

Array133. Дано множество A из N точек (точки заданы своими координатами x, y). Среди всех точек этого множества, лежащих в первой или третьей четверти, найти точку, наиболее близкую к началу координат. Если таких точек нет, то вывести точку с нулевыми координатами.

Array134°. Дано множество A из N точек (точки заданы своими координатами x, y). Найти пару точек этого множества с максимальным расстоянием между ними и само это расстояние (точки выводятся в том же порядке, в котором они перечислены при задании множества A).

Array135. Даны множества A и B, состоящие соответственно из N_1 и N_2 точек (точки заданы своими координатами x, y). Найти минимальное расстояние между точками этих множеств и сами точки, расположенные на этом расстоянии (вначале выводится точка из множества A, затем точка из множества B).

Array136. Дано множество A из N точек ($N > 2$, точки заданы своими координатами x, y). Найти такую точку из данного множества, сумма расстояний от которой до остальных его точек минимальна, и саму эту сумму.

Array137. Дано множество A из N точек ($N > 2$, точки заданы своими координатами x, y). Найти наибольший периметр треугольника, вершины которого принадлежат различным точкам множества A, и сами эти точки (точки выводятся в том же порядке, в котором они перечислены при задании множества A).

Array138. Дано множество A из N точек ($N > 2$, точки заданы своими координатами x, y). Найти наименьший периметр треугольника, вершины которого принадлежат различным точкам множества A, и сами эти точки (точки выводятся в том же порядке, в котором они перечислены при задании множества A).

Array139. Дано множество A из N точек с целочисленными координатами x, y.

Порядок на координатной плоскости определим следующим образом:

$$(x_1, y_1) < (x_2, y_2), \quad \text{если либо } x_1 < x_2, \text{ либо } x_1 = x_2 \text{ и } y_1 < y_2.$$

Расположить точки данного множества по возрастанию в соответствии с указанным порядком.

Array140. Дано множество A из N точек с целочисленными координатами x, y.

Порядок на координатной плоскости определим следующим образом:

$$(x_1, y_1) < (x_2, y_2), \quad \text{если либо } x_1 + y_1 < x_2 + y_2, \text{ либо } x_1 + y_1 = x_2 + y_2 \text{ и } x_1 < x_2.$$

Расположить точки данного множества по убыванию в соответствии с указанным порядком.

Двумерные массивы (матрицы)

Условие вида «дана матрица размера $M \times N$» означает, что вначале дается *фактический размер* двумерного массива-матрицы (количество строк M и количество столбцов N), а затем приводятся элементы этого массива (количество элементов равно $M \cdot N$). Если в задании явно не указывается, какие значения могут принимать размеры исходной матрицы, то предполагается, что и число строк, и число столбцов может изменяться в пределах от 2 до 10. Начальные значения как первого, так и второго индекса двумерного массива-матрицы всегда считаются равными 1. Ввод и вывод элементов матрицы осуществляются *по строкам*.

Квадратной матрицей порядка M называется двумерный массив-матрица размера $M \times M$.

Если в задании, связанном с созданием или преобразованием матрицы, не описан результирующий набор данных, то предполагается, что этим набором является созданная (преобразованная) матрица, и необходимо вывести все ее элементы.

Формирование матрицы и вывод ее элементов

В заданиях на формирование матрицы предполагается, что размер результирующей матрицы не превосходит 10×10.

Matrix1. Даны целые положительные числа M и N. Сформировать целочисленную матрицу размера $M \times N$, у которой все элементы I-й строки имеют значение $10 \cdot I$ ($I = 1, \ldots, M$).

Matrix2. Даны целые положительные числа M и N. Сформировать целочисленную матрицу размера $M \times N$, у которой все элементы J-го столбца имеют значение $5 \cdot J$ ($J = 1, \ldots, N$).

Matrix3. Даны целые положительные числа M, N и набор из M чисел. Сформировать матрицу размера $M \times N$, у которой в каждом столбце содержатся все числа из исходного набора (в том же порядке).

Matrix4. Даны целые положительные числа M, N и набор из N чисел. Сформировать матрицу размера $M \times N$, у которой в каждой строке содержатся все числа из исходного набора (в том же порядке).

Matrix5. Даны целые положительные числа M, N, число D и набор из M чисел. Сформировать матрицу размера $M \times N$, у которой первый столбец совпадает с исходным набором чисел, а элементы каждого следующего столбца равны сумме соответствующего элемента предыдущего столбца и числа D (в результате каждая строка матрицы будет содержать элементы *арифметической прогрессии*).

Matrix6. Даны целые положительные числа M, N, число D и набор из N чисел. Сформировать матрицу размера $M \times N$, у которой первая строка совпадает с исходным набором чисел, а элементы каждой следующей строки равны соответствующему элементу предыдущей строки, умноженному на D (в результате каждый столбец матрицы будет содержать элементы *геометрической прогрессии*).

Matrix7°. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le M$). Вывести элементы K-й строки данной матрицы.

Matrix8. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le N$). Вывести элементы K-го столбца данной матрицы.

Matrix9. Дана матрица размера $M \times N$. Вывести ее элементы, расположенные в строках с четными номерами (2, 4, ...). Вывод элементов производить по строкам, условный оператор не использовать.

Matrix10. Дана матрица размера $M \times N$. Вывести ее элементы, расположенные в столбцах с нечетными номерами (1, 3, ...). Вывод элементов производить по столбцам, условный оператор не использовать.

Matrix11. Дана матрица размера $M \times N$. Вывести ее элементы в следующем порядке: первая строка слева направо, вторая строка справа налево, третья

строка слева направо, четвертая строка справа налево и т. д.

Matrix12. Дана матрица размера $M \times N$. Вывести ее элементы в следующем порядке: первый столбец сверху вниз, второй столбец снизу вверх, третий столбец сверху вниз, четвертый столбец снизу вверх и т. д.

Matrix13. Дана квадратная матрица A порядка M. Начиная с элемента $A_{1,1}$, вывести ее элементы следующим образом («*уголками*»): все элементы первой строки; элементы последнего столбца, кроме первого (уже выведенного) элемента; оставшиеся элементы второй строки; оставшиеся элементы предпоследнего столбца и т. д.; последним выводится элемент $A_{M,1}$.

Matrix14. Дана квадратная матрица A порядка M. Начиная с элемента $A_{1,1}$, вывести ее элементы следующим образом («*уголками*»): все элементы первого столбца; элементы последней строки, кроме первого (уже выведенного) элемента; оставшиеся элементы второго столбца; оставшиеся элементы предпоследней строки и т. д.; последним выводится элемент $A_{1,M}$.

Matrix15. Дана квадратная матрица A порядка M (M — нечетное число). Начиная с элемента $A_{1,1}$ и перемещаясь по часовой стрелке, вывести все ее элементы *по спирали*: первая строка, последний столбец, последняя строка в обратном порядке, первый столбец в обратном порядке, оставшиеся элементы второй строки и т. д.; последним выводится центральный элемент матрицы.

Matrix16. Дана квадратная матрица A порядка M (M — нечетное число). Начиная с элемента $A_{1,1}$ и перемещаясь против часовой стрелки, вывести все ее элементы *по спирали*: первый столбец, последняя строка, последний столбец в обратном порядке, первая строка в обратном порядке, оставшиеся элементы второго столбца и т. д.; последним выводится центральный элемент матрицы.

Анализ элементов матрицы

Matrix17. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le M$). Найти сумму и произведение элементов K-й строки данной матрицы.

Matrix18. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le N$). Найти сумму и произведение элементов K-го столбца данной матрицы.

Matrix19. Дана матрица размера $M \times N$. Для каждой строки матрицы найти сумму ее элементов.

Matrix20. Дана матрица размера $M \times N$. Для каждого столбца матрицы найти произведение его элементов.

Matrix21. Дана матрица размера $M \times N$. Для каждой строки матрицы с нечетным номером (1, 3, ...) найти среднее арифметическое ее элементов. Условный оператор не использовать.

Matrix22. Дана матрица размера $M \times N$. Для каждого столбца матрицы с четным номером (2, 4, ...) найти сумму его элементов. Условный оператор не использовать.

Matrix23. Дана матрица размера $M \times N$. В каждой строке матрицы найти минимальный элемент.

Matrix24°. Дана матрица размера $M \times N$. В каждом столбце матрицы найти максимальный элемент.

Matrix25. Дана матрица размера $M \times N$. Найти номер ее строки с наибольшей суммой элементов и вывести данный номер, а также значение наибольшей суммы.

Matrix26. Дана матрица размера $M \times N$. Найти номер ее столбца с наименьшим произведением элементов и вывести данный номер, а также значение наименьшего произведения.

Matrix27. Дана матрица размера $M \times N$. Найти максимальный среди минимальных элементов ее строк.

Matrix28. Дана матрица размера $M \times N$. Найти минимальный среди максимальных элементов ее столбцов.

Matrix29. Дана матрица размера $M \times N$. В каждой ее строке найти количество элементов, меньших среднего арифметического всех элементов этой строки.

Matrix30. Дана матрица размера $M \times N$. В каждом ее столбце найти количество элементов, больших среднего арифметического всех элементов этого столбца.

Matrix31. Дана матрица размера $M \times N$. Найти номера строки и столбца для элемента матрицы, наиболее близкого к среднему значению всех ее элементов.

Matrix32. Дана целочисленная матрица размера $M \times N$. Найти номер первой из ее строк, содержащих равное количество положительных и отрицательных элементов (нулевые элементы матрицы не учитываются). Если таких строк нет, то вывести 0.

Matrix33. Дана целочисленная матрица размера $M \times N$. Найти номер последнего из ее столбцов, содержащих равное количество положительных и отрицательных элементов (нулевые элементы матрицы не учитываются).

Если таких столбцов нет, то вывести 0.

Matrix34. Дана целочисленная матрица размера $M \times N$. Найти номер последней из ее строк, содержащих только четные числа. Если таких строк нет, то вывести 0.

Matrix35. Дана целочисленная матрица размера $M \times N$. Найти номер первого из ее столбцов, содержащих только нечетные числа. Если таких столбцов нет, то вывести 0.

Matrix36°. Дана целочисленная матрица размера $M \times N$, элементы которой могут принимать значения от 0 до 100. Различные строки матрицы назовем *похожими*, если совпадают множества чисел, встречающихся в этих строках. Найти количество строк, похожих на первую строку данной матрицы.

Matrix37. Дана целочисленная матрица размера $M \times N$, элементы которой могут принимать значения от 0 до 100. Различные столбцы матрицы назовем *похожими*, если совпадают множества чисел, встречающихся в этих столбцах. Найти количество столбцов, похожих на последний столбец данной матрицы.

Matrix38. Дана целочисленная матрица размера $M \times N$. Найти количество ее строк, все элементы которых различны.

Matrix39. Дана целочисленная матрица размера $M \times N$. Найти количество ее столбцов, все элементы которых различны.

Matrix40. Дана целочисленная матрица размера $M \times N$. Найти номер последней из ее строк, содержащих максимальное количество одинаковых элементов.

Matrix41. Дана целочисленная матрица размера $M \times N$. Найти номер первого из ее столбцов, содержащих максимальное количество одинаковых элементов.

Matrix42. Дана матрица размера $M \times N$. Найти количество ее строк, элементы которых упорядочены по возрастанию.

Matrix43. Дана матрица размера $M \times N$. Найти количество ее столбцов, элементы которых упорядочены по убыванию.

Matrix44. Дана матрица размера $M \times N$. Найти минимальный среди элементов тех строк, которые упорядочены либо по возрастанию, либо по убыванию. Если упорядоченные строки в матрице отсутствуют, то вывести 0.

Matrix45. Дана матрица размера $M \times N$. Найти максимальный среди элементов тех столбцов, которые упорядочены либо по возрастанию, либо

по убыванию. Если упорядоченные столбцы в матрице отсутствуют, то вывести 0.

Matrix46. Дана целочисленная матрица размера $M \times N$. Найти элемент, являющийся максимальным в своей строке и минимальным в своем столбце. Если такой элемент отсутствует, то вывести 0.

Преобразование матрицы

При выполнении заданий из данного пункта (за исключением Matrix74 и Matrix75) не следует использовать вспомогательные двумерные массивы-матрицы.

Matrix47. Дана матрица размера $M \times N$ и целые числа K_1 и K_2 ($1 \le K_1 < K_2 \le M$). Поменять местами строки матрицы с номерами K_1 и K_2.

Matrix48. Дана матрица размера $M \times N$ и целые числа K_1 и K_2 ($1 \le K_1 < K_2 \le N$). Поменять местами столбцы матрицы с номерами K_1 и K_2.

Matrix49. Дана матрица размера $M \times N$. Преобразовать матрицу, поменяв местами минимальный и максимальный элемент в каждой строке.

Matrix50. Дана матрица размера $M \times N$. Преобразовать матрицу, поменяв местами минимальный и максимальный элемент в каждом столбце.

Matrix51. Дана матрица размера $M \times N$. Поменять местами строки, содержащие минимальный и максимальный элементы матрицы.

Matrix52. Дана матрица размера $M \times N$. Поменять местами столбцы, содержащие минимальный и максимальный элементы матрицы.

Matrix53°. Дана матрица размера $M \times N$. Поменять местами столбец с номером 1 и последний из столбцов, содержащих только положительные элементы. Если требуемых столбцов нет, то вывести матрицу без изменений.

Matrix54. Дана матрица размера $M \times N$. Поменять местами столбец с номером N и первый из столбцов, содержащих только отрицательные элементы. Если требуемых столбцов нет, то вывести матрицу без изменений.

Matrix55. Дана матрица размера $M \times N$ (M — четное число). Поменять местами верхнюю и нижнюю половины матрицы.

Matrix56. Дана матрица размера $M \times N$ (N — четное число). Поменять местами левую и правую половины матрицы.

Matrix57. Дана матрица размера $M \times N$ (M и N — четные числа). Поменять местами левую верхнюю и правую нижнюю четверти матрицы.

Matrix58. Дана матрица размера $M \times N$ (M и N — четные числа). Поменять местами левую нижнюю и правую верхнюю четверти матрицы.

Matrix59. Дана матрица размера $M \times N$. Зеркально отразить ее элементы относительно горизонтальной оси симметрии матрицы (при этом поменяются местами строки с номерами 1 и M, 2 и $M-1$ и т. д.).

Matrix60. Дана матрица размера $M \times N$. Зеркально отразить ее элементы относительно вертикальной оси симметрии матрицы (при этом поменяются местами столбцы с номерами 1 и N, 2 и $N-1$ и т. д.).

Matrix61. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le M$). Удалить строку матрицы с номером K.

Matrix62. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le N$). Удалить столбец матрицы с номером K.

Matrix63. Дана матрица размера $M \times N$. Удалить строку, содержащую минимальный элемент матрицы.

Matrix64. Дана матрица размера $M \times N$. Удалить столбец, содержащий максимальный элемент матрицы.

Matrix65. Дана матрица размера $M \times N$. Удалить ее первый столбец, содержащий только положительные элементы. Если требуемых столбцов нет, то вывести матрицу без изменений.

Matrix66. Дана матрица размера $M \times N$. Удалить ее последний столбец, содержащий только отрицательные элементы. Если требуемых столбцов нет, то вывести матрицу без изменений.

Matrix67. Дана матрица размера $M \times N$, содержащая как положительные, так и отрицательные элементы. Удалить все ее столбцы, содержащие только положительные элементы. Если требуемых столбцов нет, то вывести матрицу без изменений.

Matrix68. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le M$). Перед строкой матрицы с номером K вставить строку из нулей.

Matrix69. Дана матрица размера $M \times N$ и целое число K ($1 \le K \le N$). После столбца матрицы с номером K вставить столбец из единиц.

Matrix70. Дана матрица размера $M \times N$. Продублировать строку матрицы, содержащую ее максимальный элемент.

Matrix71. Дана матрица размера $M \times N$. Продублировать столбец матрицы, содержащий ее минимальный элемент.

Matrix72. Дана матрица размера $M \times N$. Перед первым столбцом, содержащим только положительные элементы, вставить столбец из единиц. Если требуемых столбцов нет, то вывести матрицу без изменений.

Matrix73. Дана матрица размера $M \times N$. После последнего столбца, содержащего только отрицательные элементы, вставить столбец из нулей. Если требуемых столбцов нет, то вывести матрицу без изменений.

Matrix74°. Дана матрица размера $M \times N$. Элемент матрицы называется ее *локальным минимумом*, если он меньше всех окружающих его элементов. Заменить все локальные минимумы данной матрицы на нули. При решении допускается использовать вспомогательную матрицу.

Matrix75. Дана матрица размера $M \times N$. Элемент матрицы называется ее *локальным максимумом*, если он больше всех окружающих его элементов. Поменять знак всех локальных максимумов данной матрицы на противоположный. При решении допускается использовать вспомогательную матрицу.

Matrix76. Дана матрица размера $M \times N$. Упорядочить ее строки так, чтобы их первые элементы образовывали возрастающую последовательность.

Matrix77. Дана матрица размера $M \times N$. Упорядочить ее столбцы так, чтобы их последние элементы образовывали убывающую последовательность.

Matrix78. Дана матрица размера $M \times N$. Упорядочить ее строки так, чтобы их минимальные элементы образовывали убывающую последовательность.

Matrix79. Дана матрица размера $M \times N$. Упорядочить ее столбцы так, чтобы их максимальные элементы образовывали возрастающую последовательность.

Диагонали квадратной матрицы

Matrix80. Дана квадратная матрица A порядка M. Найти сумму элементов ее *главной диагонали*, то есть диагонали, содержащей следующие элементы:

$$A_{1,1}, \quad A_{2,2}, \quad A_{3,3}, \quad \ldots, \quad A_{M,M}.$$

Matrix81. Дана квадратная матрица A порядка M. Найти среднее арифметическое элементов ее *побочной диагонали*, то есть диагонали, содержащей следующие элементы:

$$A_{1,M}, \quad A_{2,M-1}, \quad A_{3,M-2}, \quad \ldots, \quad A_{M,1}.$$

Matrix82°. Дана квадратная матрица A порядка M. Найти сумму элементов каждой ее диагонали, параллельной главной (начиная с одноэлементной диагонали $A_{1,M}$).

Matrix83. Дана квадратная матрица A порядка M. Найти сумму элементов каждой ее диагонали, параллельной побочной (начиная с одноэлементной диагонали $A_{1,1}$).

Matrix84. Дана квадратная матрица A порядка M. Найти среднее арифметическое элементов каждой ее диагонали, параллельной главной (начиная с одноэлементной диагонали $A_{1,M}$).

Matrix85. Дана квадратная матрица A порядка M. Найти среднее арифметическое элементов каждой ее диагонали, параллельной побочной (начиная с одноэлементной диагонали $A_{1,1}$).

Matrix86. Дана квадратная матрица A порядка M. Найти минимальный элемент для каждой ее диагонали, параллельной главной (начиная с одноэлементной диагонали $A_{1,M}$).

Matrix87. Дана квадратная матрица A порядка M. Найти максимальный элемент для каждой ее диагонали, параллельной побочной (начиная с одноэлементной диагонали $A_{1,1}$).

Matrix88°. Дана квадратная матрица порядка M. Обнулить элементы матрицы, лежащие ниже главной диагонали. Условный оператор не использовать.

Matrix89. Дана квадратная матрица порядка M. Обнулить элементы матрицы, лежащие выше побочной диагонали. Условный оператор не использовать.

Matrix90. Дана квадратная матрица порядка M. Обнулить элементы матрицы, лежащие на побочной диагонали и ниже нее. Условный оператор не использовать.

Matrix91. Дана квадратная матрица порядка M. Обнулить элементы матрицы, лежащие на главной диагонали и выше нее. Условный оператор не использовать.

Matrix92. Дана квадратная матрица порядка M. Обнулить элементы матрицы, лежащие одновременно выше главной диагонали и выше побочной диагонали. Условный оператор не использовать.

Matrix93. Дана квадратная матрица порядка M. Обнулить элементы матрицы, лежащие одновременно выше главной диагонали и ниже побочной диагонали. Условный оператор не использовать.

Matrix94. Дана квадратная матрица порядка M. Обнулить элементы матрицы, лежащие одновременно ниже главной диагонали (включая эту диагональ) и выше побочной диагонали (также включая эту диагональ). Условный оператор не использовать.

Matrix95. Дана квадратная матрица порядка M. Обнулить элементы матрицы,

лежащие одновременно ниже главной диагонали (включая эту диагональ) и ниже побочной диагонали (также включая эту диагональ). Условный оператор не использовать.

Matrix96. Дана квадратная матрица A порядка M. Зеркально отразить ее элементы относительно главной диагонали (при этом элементы главной диагонали останутся на прежнем месте, элемент $A_{1,2}$ поменяется местами с $A_{2,1}$, элемент $A_{1,3}$ — с $A_{3,1}$ и т. д.). Вспомогательную матрицу не использовать.

Matrix97. Дана квадратная матрица A порядка M. Зеркально отразить ее элементы относительно побочной диагонали (при этом элементы побочной диагонали останутся на прежнем месте, элемент $A_{1,1}$ поменяется местами с $A_{M,M}$, элемент $A_{1,2}$ — с $A_{M-1,M}$ и т. д.). Вспомогательную матрицу не использовать.

Matrix98. Дана квадратная матрица A порядка M. Повернуть ее на угол 180° (при этом элемент $A_{1,1}$ поменяется местами с $A_{M,M}$, элемент $A_{1,2}$ — с $A_{M,M-1}$ и т. д.). Вспомогательную матрицу не использовать.

Matrix99. Дана квадратная матрица A порядка M. Повернуть ее на угол 90° в положительном направлении, то есть против часовой стрелки (при этом элемент $A_{1,1}$ перейдет в $A_{M,1}$, элемент $A_{M,1}$ — в $A_{M,M}$ и т. д.). Вспомогательную матрицу не использовать.

Matrix100°. Дана квадратная матрица A порядка M. Повернуть ее на угол 90° в отрицательном направлении, то есть по часовой стрелке (при этом элемент $A_{1,1}$ перейдет в $A_{1,M}$, элемент $A_{1,M}$ — в $A_{M,M}$ и т. д.). Вспомогательную матрицу не использовать.

Символы и строки

При выполнении заданий на обработку русских букв можно считать, что буква «ё» в исходных строковых данных отсутствует.

Символы и их коды. Формирование строк

String1. Дан символ C. Вывести его *код* (то есть номер в кодовой таблице).

String2. Дано целое число N ($32 \le N \le 126$). Вывести символ с кодом, равным N.

String3. Дан символ C. Вывести два символа, первый из которых предшествует символу C в кодовой таблице, а второй следует за символом C.

String4. Дано целое число N ($1 \le N \le 26$). Вывести N первых *прописных* (то есть заглавных) букв латинского алфавита.

String5. Дано целое число N ($1 \le N \le 26$). Вывести N последних *строчных* (то есть маленьких) букв латинского алфавита в обратном порядке (начиная с буквы «z»).

String6. Дан символ C, изображающий цифру или букву (латинскую или русскую). Если C изображает цифру, то вывести строку «digit», если латинскую букву — вывести строку «lat», если русскую — вывести строку «rus».

String7. Дана непустая строка. Вывести коды ее первого и последнего символа.

String8. Дано целое число N (>0) и символ C. Вывести строку длины N, которая состоит из символов C.

String9°. Дано четное число N (>0) и символы C_1 и C_2. Вывести строку длины N, которая состоит из чередующихся символов C_1 и C_2, начиная с C_1.

String10°. Дана строка. Вывести строку, содержащую те же символы, но расположенные в обратном порядке.

String11. Дана непустая строка S. Вывести строку, содержащую символы строки S, между которыми вставлено по одному пробелу.

String12. Дана непустая строка S и целое число N (>0). Вывести строку, содержащую символы строки S, между которыми вставлено по N символов «*» (звездочка).

Посимвольный анализ и преобразование строк. Строки и числа

String13. Дана строка. Подсчитать количество содержащихся в ней цифр.

String14. Дана строка. Подсчитать количество содержащихся в ней прописных латинских букв.

String15. Дана строка. Подсчитать общее количество содержащихся в ней строчных латинских и русских букв.

String16. Дана строка. Преобразовать в ней все прописные латинские буквы в строчные.

String17. Дана строка. Преобразовать в ней все строчные буквы (как латинские, так и русские) в прописные.

String18. Дана строка. Преобразовать в ней все строчные буквы (как латинские, так и русские) в прописные, а прописные — в строчные.

String19°. Дана строка. Если она представляет собой запись целого числа, то вывести 1, если вещественного (с дробной частью) — вывести 2; если строку нельзя преобразовать в число, то вывести 0. Считать, что дробная часть вещественного числа отделяется от его целой части десятичной *точкой* «.».

String20. Дано целое положительное число. Вывести символы, изображающие цифры этого числа (в порядке слева направо).

String21. Дано целое положительное число. Вывести символы, изображающие цифры этого числа (в порядке справа налево).

String22. Дана строка, изображающая целое положительное число. Вывести сумму цифр этого числа.

String23. Дана строка, изображающая арифметическое выражение вида «<цифра>±<цифра>±...±<цифра>», где на месте знака операции «±» находится символ «+» или «−» (например, «4+7−2−8»). Вывести значение данного выражения (целое число).

String24. Дана строка, изображающая двоичную запись целого положительного числа. Вывести строку, изображающую десятичную запись этого же числа.

String25. Дана строка, изображающая десятичную запись целого положительного числа. Вывести строку, изображающую двоичную запись этого же числа.

Обработка строк с помощью стандартных функций. Поиск и замена

В заданиях, связанных с поиском и заменой подстрок, можно считать, что исходная строка не содержит *перекрывающихся* вхождений требуемых подстрок. В заданиях String32, String35 и String38, кроме этого, можно также считать, что удаление (в String32 и String35) или замена (в String38) любого вхождения подстроки не приведет к появлению в строке *новых* вхождений данной подстроки.

String26. Дано целое число N (>0) и строка S. Преобразовать строку S в строку длины N следующим образом: если длина строки S больше N, то

отбросить первые символы, если длина строки S меньше N, то в ее начало добавить символы «.» (точка).

String27. Даны целые положительные числа N_1 и N_2 и строки S_1 и S_2. Получить из этих строк новую строку, содержащую первые N_1 символов строки S_1 и последние N_2 символов строки S_2 (в указанном порядке).

String28. Дан символ C и строка S. Удвоить каждое вхождение символа C в строку S.

String29°. Дан символ C и строки S, S_0. Перед каждым вхождением символа C в строку S вставить строку S_0.

String30. Дан символ C и строки S, S_0. После каждого вхождения символа C в строку S вставить строку S_0.

String31. Даны строки S и S_0. Проверить, содержится ли строка S_0 в строке S. Если содержится, то вывести TRUE, если не содержится, то вывести FALSE.

String32. Даны строки S и S_0. Найти количество вхождений строки S_0 в строку S.

String33. Даны строки S и S_0. Удалить из строки S первую подстроку, совпадающую с S_0. Если совпадающих подстрок нет, то вывести строку S без изменений.

String34. Даны строки S и S_0. Удалить из строки S последнюю подстроку, совпадающую с S_0. Если совпадающих подстрок нет, то вывести строку S без изменений.

String35. Даны строки S и S_0. Удалить из строки S все подстроки, совпадающие с S_0. Если совпадающих подстрок нет, то вывести строку S без изменений.

String36. Даны строки S, S_1 и S_2. Заменить в строке S первое вхождение строки S_1 на строку S_2.

String37. Даны строки S, S_1 и S_2. Заменить в строке S последнее вхождение строки S_1 на строку S_2.

String38. Даны строки S, S_1 и S_2. Заменить в строке S все вхождения строки S_1 на строку S_2.

String39. Дана строка, содержащая по крайней мере один символ пробела. Вывести подстроку, расположенную между первым и вторым пробелом исходной строки. Если строка содержит только один пробел, то вывести пустую строку.

String40. Дана строка, содержащая по крайней мере один символ пробела. Вывести подстроку, расположенную между первым и последним пробелом

исходной строки. Если строка содержит только один пробел, то вывести пустую строку.

Анализ и преобразование слов в строке

Во всех заданиях данного пункта предполагается, что исходные строки являются непустыми и не содержат начальных и конечных пробелов.

String41°. Дана строка, состоящая из русских слов, разделенных пробелами (одним или несколькими). Найти количество слов в строке.

String42. Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Найти количество слов, которые начинаются и заканчиваются одной и той же буквой.

String43. Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Найти количество слов, которые содержат хотя бы одну букву «А».

String44°. Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Найти количество слов, которые содержат ровно три буквы «А».

String45. Дана строка, состоящая из русских слов, разделенных пробелами (одним или несколькими). Найти длину самого короткого слова.

String46. Дана строка, состоящая из русских слов, разделенных пробелами (одним или несколькими). Найти длину самого длинного слова.

String47. Дана строка, состоящая из русских слов, разделенных пробелами (одним или несколькими). Вывести строку, содержащую эти же слова, разделенные одним символом «.» (точка). В конце строки точку не ставить.

String48. Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Преобразовать каждое слово в строке, заменив в нем все последующие вхождения его первой буквы на символ «.» (точка). Например, слово «МИНИМУМ» надо преобразовать в «МИНИ.У.». Количество пробелов между словами не изменять.

String49. Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Преобразовать каждое слово в строке, заменив в нем все предыдущие вхождения его последней буквы на символ «.» (точка). Например, слово «МИНИМУМ»

надо преобразовать в «.ИНИ.УМ». Количество пробелов между словами не изменять.

String50. Дана строка, состоящая из русских слов, разделенных пробелами (одним или несколькими). Вывести строку, содержащую эти же слова, разделенные одним пробелом и расположенные в обратном порядке.

String51. Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Вывести строку, содержащую эти же слова, разделенные одним пробелом и расположенные в алфавитном порядке.

String52. Дана строка-предложение на русском языке. Преобразовать строку так, чтобы каждое слово начиналось с заглавной буквы. *Словом* считать набор символов, не содержащий пробелов и ограниченный пробелами или началом/концом строки. Слова, не начинающиеся с буквы, не изменять.

String53. Дана строка-предложение на русском языке. Подсчитать количество содержащихся в строке знаков препинания.

String54. Дана строка-предложение на русском языке. Подсчитать количество содержащихся в строке гласных букв.

String55. Дана строка-предложение на русском языке. Вывести самое длинное слово в предложении. Если таких слов несколько, то вывести первое из них. *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки.

String56. Дана строка-предложение на русском языке. Вывести самое короткое слово в предложении. Если таких слов несколько, то вывести последнее из них. *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки.

String57. Дана строка-предложение с избыточными пробелами между словами. Преобразовать ее так, чтобы между словами был ровно один пробел.

Дополнительные задания на обработку строк

String58. Дана строка, содержащая *полное имя файла*, то есть имя диска, список каталогов (путь), собственно имя и расширение. Выделить из этой строки имя файла (без расширения).

String59. Дана строка, содержащая *полное имя файла*, то есть имя диска, список каталогов (путь), собственно имя и расширение. Выделить из этой

строки расширение файла (без предшествующей точки).

String60. Дана строка, содержащая полное имя файла. Выделить из этой строки название первого каталога (без символов «\»). Если файл содержится в корневом каталоге, то вывести символ «\».

String61. Дана строка, содержащая полное имя файла. Выделить из этой строки название последнего каталога (без символов «\»). Если файл содержится в корневом каталоге, то вывести символ «\».

String62. Дана строка-предложение на русском языке. Зашифровать ее, выполнив циклическую замену каждой буквы на следующую за ней в алфавите и сохранив при этом регистр букв («А» перейдет в «Б», «а» — в «б», «Б» — в «В», «я» — в «а» и т. д.). Букву «ё» в алфавите не учитывать («е» должна переходить в «ж»). Знаки препинания и пробелы не изменять.

String63°. Дана строка-предложение на русском языке и число K ($0 < K < 10$). Зашифровать строку, выполнив циклическую замену каждой буквы на букву того же регистра, расположенную в алфавите на K-й позиции после шифруемой буквы (например, для $K = 2$ «А» перейдет в «В», «а» — в «в», «Б» — в «Г», «я» — в «б» и т. д.). Букву «ё» в алфавите не учитывать, знаки препинания и пробелы не изменять.

String64. Дано зашифрованное предложение на русском языке (способ шифрования описан в задании String63) и кодовое смещение K ($0 < K < 10$). Расшифровать предложение.

String65. Дано зашифрованное предложение на русском языке (способ шифрования описан в задании String63) и его расшифрованный первый символ C. Найти кодовое смещение K и расшифровать предложение.

String66. Дана строка-предложение. Зашифровать ее, поместив вначале все символы, расположенные на четных позициях строки, а затем, в обратном порядке, все символы, расположенные на нечетных позициях (например, строка «Программа» превратится в «ргамамроП»).

String67. Дано предложение, зашифрованное по правилу, описанному в задании String66. Расшифровать это предложение.

String68. Дана строка, содержащая цифры и строчные латинские буквы. Если буквы в строке упорядочены по алфавиту, то вывести 0; в противном случае вывести номер первого символа строки, нарушающего алфавитный порядок.

String69. Дана строка, содержащая латинские буквы и круглые скобки. Если скобки расставлены правильно (то есть каждой открывающей соот-

ветствует одна закрывающая), то вывести число 0. В противном случае вывести или номер позиции, в которой расположена первая ошибочная закрывающая скобка, или, если закрывающих скобок не хватает, число -1.

String70°. Дана строка, содержащая латинские буквы и скобки трех видов: «()», «[]», «{}». Если скобки расставлены правильно (то есть каждой открывающей соответствует закрывающая скобка того же вида), то вывести число 0. В противном случае вывести или номер позиции, в которой расположена первая ошибочная скобка, или, если закрывающих скобок не хватает, число -1.

Двоичные файлы

Условие вида «дан файл (целых чисел, вещественных чисел и т. д.)» означает, что в наборе исходных данных указано *имя* данного файла (текстовая строка), а сам исходный файл существует и находится в текущем каталоге. Если существование исходных файлов требуется проверять в ходе выполнения задания, то это особо оговаривается в формулировке задания. Элементы исходных файлов следует считывать с помощью стандартных процедур используемого языка программирования.

Если в задании требуется создать новый файл, то имя создаваемого файла также входит в набор исходных данных и обычно является *последним* элементом этого набора. Записывать элементы в создаваемые (или модифицируемые) файлы следует с помощью стандартных процедур используемого языка программирования.

Под *размером* типизированного файла всегда подразумевается количество содержащихся в нем *элементов* указанного типа (а не количество байтов, как это принято в операционной системе). В заданиях предполагается, что файловые элементы нумеруются от 1, поэтому в языке Pascal для перехода к K-му элементу файла F необходимо использовать процедуру Seek(F, $K - 1$).

Если о минимальном размере исходного файла в задании ничего не сказано, то предполагается, что он равен 2 (то есть файл содержит по крайней мере два элемента). Максимальный размер исходных файлов не устанавливается, поэтому при решении заданий не следует использовать вспомогательные массивы, содержащие все элементы исходных файлов, однако допускается использование *вспомогательных файлов*.

Основные операции с двоичными файлами

Создание файла, ввод и вывод его элементов

File1. Дана строка S. Если S является допустимым именем файла, то создать пустой файл с этим именем и вывести TRUE. Если файл с именем S создать нельзя, то вывести FALSE.

File2°. Дано имя файла и целое число N (>1). Создать файл целых чисел с данным именем и записать в него N первых положительных четных чисел (2, 4, . . .).

File3. Дано имя файла и вещественные числа A и D. Создать файл вещественных чисел с данным именем и записать в него 10 первых членов *арифметической прогрессии* с начальным членом A и разностью D:

$$A, \quad A + D, \quad A + 2 \cdot D, \quad A + 3 \cdot D, \quad \ldots .$$

File4. Даны имена четырех файлов. Найти количество файлов с указанными именами, которые имеются в текущем каталоге.

File5. Дано имя файла целых чисел. Найти количество элементов, содержащихся в данном файле. Если файла с таким именем не существует, то вывести -1.

File6. Дано целое число K и файл, содержащий неотрицательные целые числа. Вывести K-й элемент файла (элементы нумеруются от 1). Если такой элемент отсутствует, то вывести -1.

File7. Дан файл целых чисел, содержащий не менее четырех элементов. Вывести первый, второй, предпоследний и последний элементы данного файла.

File8. Даны имена двух файлов вещественных чисел. Известно, что первый из них существует и является непустым, а второй в текущем каталоге отсутствует. Создать отсутствующий файл и записать в него начальный и конечный элементы существующего файла (в указанном порядке).

File9. Даны имена двух файлов вещественных чисел. Известно, что один из них (не обязательно первый) существует и является непустым, а другой в текущем каталоге отсутствует. Создать отсутствующий файл и записать в него конечный и начальный элементы существующего файла (в указанном порядке).

File10°. Дан файл целых чисел. Создать новый файл, содержащий те же элементы, что и исходный файл, но в обратном порядке.

File11. Дан файл вещественных чисел. Создать два новых файла, первый из которых содержит элементы исходного файла с нечетными номерами

(1, 3, . . .), а второй — с четными (2, 4, . . .).

File12. Дан файл целых чисел. Создать два новых файла, первый из которых содержит четные числа из исходного файла, а второй — нечетные (в том же порядке). Если четные или нечетные числа в исходном файле отсутствуют, то соответствующий результирующий файл оставить пустым.

File13. Дан файл целых чисел. Создать два новых файла, первый из которых содержит положительные числа из исходного файла (в обратном порядке), а второй — отрицательные (также в обратном порядке). Если положительные или отрицательные числа в исходном файле отсутствуют, то соответствующий результирующий файл оставить пустым.

File14. Дан файл вещественных чисел. Найти среднее арифметическое его элементов.

File15. Дан файл вещественных чисел. Найти сумму его элементов с четными номерами.

File16. Дан файл целых чисел. Найти количество содержащихся в нем *серий* (то есть наборов последовательно расположенных одинаковых элементов). Например, для файла с элементами 1, 5, 5, 5, 4, 4, 5 результат равен 4.

File17. Дан файл целых чисел. Создать новый файл целых чисел, содержащий длины всех серий исходного файла (*серией* называется набор последовательно расположенных одинаковых элементов, а *длиной серии* — количество этих элементов). Например, для исходного файла с элементами 1, 5, 5, 5, 4, 4, 5 содержимое результирующего файла должно быть следующим: 1, 3, 2, 1.

File18. Дан файл вещественных чисел. Найти его первый локальный минимум (*локальным минимумом* называется элемент, который меньше своих соседей).

File19. Дан файл вещественных чисел. Найти его последний локальный максимум (*локальным максимумом* называется элемент, который больше своих соседей).

File20. Дан файл вещественных чисел. Найти общее количество его *локальных экстремумов*, то есть локальных минимумов и локальных максимумов (определения локального минимума и локального максимума даны в заданиях File18 и File19).

File21. Дан файл вещественных чисел. Создать файл целых чисел, содержащий номера всех *локальных максимумов* исходного файла в порядке

возрастания (определение локального максимума дано в задании File19).

File22. Дан файл вещественных чисел. Создать файл целых чисел, содержащий номера всех *локальных экстремумов* исходного файла в порядке убывания (определение локального экстремума дано в задании File20).

File23. Дан файл вещественных чисел. Создать файл целых чисел, содержащий длины всех убывающих последовательностей элементов исходного файла. Например, для исходного файла с элементами 1.7, 4.5, 3.4, 2.2, 8.5, 1.2 содержимое результирующего файла должно быть следующим: 3, 2.

File24. Дан файл вещественных чисел. Создать файл целых чисел, содержащий длины всех монотонных последовательностей элементов исходного файла. Например, для исходного файла с элементами 1.7, 4.5, 3.4, 2.2, 8.5, 1.2 содержимое результирующего файла должно быть следующим: 2, 3, 2, 2.

Преобразование файла

File25°. Дан файл вещественных чисел. Заменить в нем все элементы на их квадраты.

File26. Дан файл вещественных чисел. Поменять в нем местами минимальный и максимальный элементы.

File27°. Дан файл целых чисел с элементами $A_1, A_2, \ldots, A_N$ (N — количество элементов в файле). Заменить исходное расположение его элементов на следующее:

$$A_1, \quad A_N, \quad A_2, \quad A_{N-1}, \quad A_3, \quad \ldots .$$

File28. Дан файл вещественных чисел. Заменить в файле каждый элемент, кроме начального и конечного, на его среднее арифметическое с предыдущим и последующим элементом.

File29. Дан файл целых чисел, содержащий более 50 элементов. Уменьшить его размер до 50 элементов, удалив из файла необходимое количество конечных элементов.

File30. Дан файл целых чисел, содержащий четное количество элементов. Удалить из данного файла вторую половину элементов.

File31. Дан файл целых чисел, содержащий более 50 элементов. Уменьшить его размер до 50 элементов, удалив из файла необходимое количество начальных элементов.

File32. Дан файл целых чисел, содержащий четное количество элементов. Удалить из данного файла первую половину элементов.

File33. Дан файл целых чисел. Удалить из него все элементы с четными номерами.

File34. Дан файл целых чисел. Удалить из него все отрицательные числа.

File35. Дан файл целых чисел, содержащий менее 50 элементов. Увеличить его размер до 50 элементов, записав в начало файла необходимое количество нулей.

File36. Дан файл целых чисел. Удвоить его размер, записав в конец файла все его исходные элементы (в том же порядке).

File37. Дан файл целых чисел. Удвоить его размер, записав в конец файла все его исходные элементы (в обратном порядке).

File38. Дан файл целых чисел. Продублировать в нем все элементы с нечетными номерами.

File39. Дан файл целых чисел. Продублировать в нем все числа, принадлежащие диапазону 5–10.

File40. Дан файл целых чисел. Заменить в нем каждый элемент с четным номером на два нуля.

File41°. Дан файл целых чисел. Заменить в нем каждое положительное число на три нуля.

Обработка нетипизированных двоичных файлов

File42. Даны два файла произвольного типа. Поменять местами их содержимое.

File43°. Дан файл произвольного типа. Создать его копию с новым именем.

File44. Даны три файла одного и того же типа, но разного размера. Заменить содержимое самого длинного файла на содержимое самого короткого.

File45. Даны три файла одного и того же типа, но разного размера. Заменить содержимое самого короткого файла на содержимое самого длинного.

File46. Дана строка S_0, целое число N (≤ 4) и N файлов одного и того же типа с именами S_1, ..., S_N. Объединить содержимое этих файлов (в указанном порядке) в новом файле с именем S_0.

File47. Даны два файла одного и того же типа. Добавить к первому файлу содержимое второго файла, а ко второму файлу — содержимое первого.

Работа с несколькими числовыми файлами. Файлы-архивы

File48°. Даны три файла целых чисел одинакового размера с именами S_A, S_B, S_C и строка S_D. Создать новый файл с именем S_D, в котором чередовались бы элементы исходных файлов с одним и тем же номером:

$$A_1, \quad B_1, \quad C_1, \quad A_2, \quad B_2, \quad C_2, \quad \ldots .$$

File49. Даны четыре файла целых чисел разного размера с именами S_A, S_B, S_C, S_D и строка S_E. Создать новый файл с именем S_E, в котором чередовались бы элементы исходных файлов с одним и тем же номером (как в задании File48). «Лишние» элементы более длинных файлов в результирующий файл не записывать.

File50°. Даны два файла вещественных чисел с именами S_1 и S_2, элементы которых упорядочены по возрастанию. Объединить эти файлы в новый файл с именем S_3 так, чтобы его элементы также оказались упорядоченными по возрастанию.

File51. Даны три файла вещественных чисел с именами S_1, S_2 и S_3, элементы которых упорядочены по убыванию. Объединить эти файлы в новый файл с именем S_4 так, чтобы его элементы также оказались упорядоченными по убыванию.

File52. Дана строка S_0, целое число N (≤ 4) и N файлов целых чисел с именами S_1, ..., S_N. Объединить их содержимое в новом файле-*архиве* с именем S_0, используя следующий формат: в первом элементе файла-архива хранится число N, в следующих N элементах хранится размер (число элементов) каждого из исходных файлов, а затем последовательно размещаются данные из каждого исходного файла.

File53. Дана строка S, целое число N (>0) и файл-*архив* целых чисел, содержащий данные из нескольких файлов в формате, описанном в задании File52. Восстановить из файла-архива файл с номером N и сохранить его под именем S. Если файл-архив содержит данные из менее чем N файлов, то оставить результирующий файл пустым.

File54. Дана строка S и файл-*архив* целых чисел, содержащий данные из нескольких (не более шести) файлов в формате, описанном в задании File52. Для каждого из файлов, содержащихся в архиве, найти среднее арифметическое всех его элементов (вещественное число) и записать найденные числа (в том же порядке) в файл вещественных чисел с именем S.

File55. Дана строка S_0, целое число N (≤ 4) и N файлов целых чисел с именами S_1, ..., S_N. Объединить их содержимое в новом файле-*архиве* с именем S_0,

последовательно записывая в него следующие данные: размер (число элементов) первого исходного файла и все элементы этого файла, размер второго исходного файла и все его элементы, ..., размер N-го исходного файла и все его элементы.

File56. Дана строка S, целое число N (>0) и файл-*архив* целых чисел, содержащий данные из нескольких файлов в формате, описанном в задании File55. Восстановить из файла-архива файл с номером N и сохранить его под именем S. Если файл-архив содержит данные из менее чем N файлов, то оставить результирующий файл пустым.

File57. Даны строки S_1, S_2 и файл-*архив* целых чисел, содержащий данные из нескольких файлов в формате, описанном в задании File55. Создать новые файлы целых чисел с именами S_1 и S_2 и записать в первый из них начальные элементы всех файлов, содержащихся в архиве, а во второй — конечные элементы этих файлов (в том же порядке).

Символьные и строковые файлы

Строковым файлом называется типизированный файл, элементами которого являются текстовые строки. В системе Borland Delphi файловые переменные для строковых файлов необходимо описывать как file of ShortString; этот же тип следует указывать при описании переменных, которые используются в процедурах ввода-вывода для строковых файлов.

File58°. Дан символьный файл, содержащий по крайней мере один символ пробела. Удалить все его элементы, расположенные после первого символа пробела, включая и этот пробел.

File59. Дан символьный файл, содержащий по крайней мере один символ пробела. Удалить все его элементы, расположенные после последнего символа пробела, включая и этот пробел.

File60. Дан символьный файл, содержащий по крайней мере один символ пробела. Удалить все его элементы, расположенные перед первым символом пробела, включая и этот пробел.

File61°. Дан символьный файл, содержащий по крайней мере один символ пробела. Удалить все его элементы, расположенные перед последним символом пробела, включая и этот пробел.

File62. Дан символьный файл. Упорядочить его элементы по возрастанию их кодов.

File63°. Дано целое число K (> 0) и строковый файл. Создать два новых файла: строковый, содержащий первые K символов каждой строки исходного файла, и символьный, содержащий K-й символ каждой строки (если длина строки меньше K, то в строковый файл записывается вся строка, а в символьный файл записывается пробел).

File64. Дан строковый файл. Создать новый строковый файл, содержащий все строки исходного файла наименьшей длины (в том же порядке).

File65. Дан строковый файл. Создать новый строковый файл, содержащий все строки исходного файла наибольшей длины (в обратном порядке).

File66. Дан строковый файл. Создать новый строковый файл, в котором строки из исходного файла располагались бы в *лексикографическом* порядке, то есть по возрастанию кодов их символов, начиная с первого символа.

File67°. Дан строковый файл, содержащий даты в формате «день/месяц/год», причем под день и месяц отводится по две позиции, а под год — четыре (например, «16/04/2001»). Создать два файла целых чисел, первый из которых содержит значения дней, а второй — значения месяцев для дат из исходного строкового файла (в том же порядке).

File68. Дан строковый файл, содержащий даты в формате, описанном в задании File67. Создать два файла целых чисел, первый из которых содержит значения месяцев, а второй — значения лет для дат из исходного строкового файла (в обратном порядке).

File69. Дан строковый файл, содержащий даты в формате, описанном в задании File67. Создать новый строковый файл, содержащий все летние даты из исходного файла (в том же порядке). Если даты с требуемым временем года в файле отсутствуют, то оставить результирующий файл пустым.

File70. Дан строковый файл, содержащий даты в формате, описанном в задании File67. Создать новый строковый файл, содержащий все зимние даты из исходного файла (в обратном порядке). Если даты с требуемым временем года в файле отсутствуют, то оставить результирующий файл пустым.

File71. Дан строковый файл, содержащий даты в формате, описанном в задании File67. Найти строку, содержащую самую раннюю весеннюю дату. Если даты с требуемым временем года в файле отсутствуют, то вывести пустую строку.

File72. Дан строковый файл, содержащий даты в формате, описанном в задании File67. Найти строку, содержащую самую позднюю осеннюю дату.

Если даты с требуемым временем года в файле отсутствуют, то вывести пустую строку.

File73. Дан строковый файл, содержащий даты в формате, описанном в задании File67. Создать новый строковый файл, в котором даты из исходного файла располагались бы в порядке убывания.

Использование файлов для работы с матрицами

Матрицей размера $M \times N$ называется прямоугольная таблица чисел, содержащая M строк и N столбцов. Для работы с матрицами обычно используются *двумерные массивы* (см. задания группы Matrix). Данный пункт посвящен способам обработки матриц, хранящихся в типизированных файлах на внешних носителях (дисках). Как и в остальных заданиях на обработку файловых данных, при выполнении заданий из данного пункта *не следует* использовать вспомогательные массивы, содержащие все файловые элементы.

В заданиях данного пункта используются дополнительные понятия теории матриц. Приведем определения этих понятий.

Пусть A — матрица размера $M \times N$. Матрица B называется *транспонированной* к матрице A, если она имеет размер $N \times M$ и ее элементы удовлетворяют следующему соотношению:

$$B_{I,J} = A_{J,I}, \qquad I = 1, \ldots, N, \quad J = 1, \ldots M.$$

Пусть A — матрица размера $M \times N$, B — матрица размера $N \times P$. Матрица C называется *произведением* матриц A и B (и обозначается $A \cdot B$), если она имеет размер $M \times P$ и ее элементы удовлетворяют следующему соотношению:

$$C_{I,J} = A_{I,1} \cdot B_{1,J} + A_{I,2} \cdot B_{2,J} + \ldots + A_{1,N} \cdot B_{N,J}, \qquad I = 1, \ldots, M, \quad J = 1, \ldots P.$$

Квадратная матрица A называется *верхнетреугольной*, если все ее элементы, лежащие ниже главной диагонали, равны нулю (определение *главной диагонали* см. в задании Matrix80):

$$A_{I,J} = 0, \qquad I > J.$$

Квадратная матрица A называется *нижнетреугольной*, если все ее элементы, лежащие выше главной диагонали, равны нулю:

$$A_{I,J} = 0, \qquad I < J.$$

Квадратная матрица A называется *трехдиагональной*, если равны нулю все ее элементы, не лежащие на главной диагонали и на двух диагоналях, примыкающих к главной:

$$A_{I,J} = 0, \qquad |I - J| > 1.$$

File74°. Даны два целых числа I, J и файл вещественных чисел, содержащий элементы квадратной матрицы (по строкам). Вывести элемент матрицы, расположенный в I-й строке и J-м столбце (строки и столбцы нумеруются от 1). Если требуемый элемент отсутствует, то вывести 0.

File75. Дан файл вещественных чисел, содержащий элементы квадратной матрицы (по строкам). Создать новый файл, содержащий элементы матрицы, транспонированной к исходной.

File76. Даны два файла вещественных чисел с именами S_A и S_B, содержащие элементы квадратных матриц A и B (по строкам). Создать новый файл с именем S_C, содержащий элементы произведения $A{\cdot}B$. Если матрицы A и B нельзя перемножать, то оставить файл S_C пустым.

File77. Даны два целых числа I, J и файл вещественных чисел, содержащий элементы прямоугольной матрицы (по строкам), причем первый элемент файла содержит количество столбцов матрицы. Вывести элемент матрицы, расположенный в I-й строке и J-м столбце (строки и столбцы нумеруются от 1). Если требуемый элемент отсутствует, то вывести 0.

File78. Дан файл вещественных чисел, содержащий элементы прямоугольной матрицы (по строкам), причем первый элемент файла содержит количество столбцов матрицы. Создать новый файл той же структуры, содержащий матрицу, транспонированную к исходной.

File79. Даны два файла вещественных чисел с именами S_A и S_B, содержащие элементы прямоугольных матриц A и B (по строкам), причем первый элемент каждого файла содержит количество столбцов соответствующей матрицы. Создать файл той же структуры с именем S_C, содержащий элементы произведения $A{\cdot}B$. Если матрицы A и B нельзя перемножать, то оставить файл S_C пустым.

File80. Дан файл вещественных чисел, содержащий элементы верхнетреугольной матрицы (по строкам). Создать новый файл, содержащий элементы ненулевой части данной матрицы (по строкам).

File81. Дан файл вещественных чисел, содержащий элементы нижнетреугольной матрицы (по строкам). Создать новый файл, содержащий элементы ненулевой части данной матрицы (по строкам).

File82. Дан файл вещественных чисел, содержащий элементы трехдиагональной матрицы (по строкам). Создать новый файл, содержащий элементы ненулевой части данной матрицы (по строкам).

File83. Даны два целых числа I, J и файл вещественных чисел, содержащий ненулевую часть верхнетреугольной матрицы (по строкам). Вывести порядок матрицы и ее элемент, расположенный в I-й строке и J-м столбце (строки и столбцы нумеруются от 1). Если требуемый элемент находится в нулевой части матрицы, то вывести 0; если элемент отсутствует, то вывести -1.

File84. Даны два целых числа I, J и файл вещественных чисел, содержащий ненулевую часть нижнетреугольной матрицы (по строкам). Вывести порядок матрицы и ее элемент, расположенный в I-й строке и J-м столбце (строки и столбцы нумеруются от 1). Если требуемый элемент находится в нулевой части матрицы, то вывести 0; если элемент отсутствует, то вывести -1.

File85. Даны два целых числа I, J и файл вещественных чисел, содержащий ненулевую часть трехдиагональной матрицы (по строкам). Вывести порядок матрицы и ее элемент, расположенный в I-й строке и J-м столбце (строки и столбцы нумеруются от 1). Если требуемый элемент находится в нулевой части матрицы, то вывести 0; если элемент отсутствует, то вывести -1.

File86. Дан файл вещественных чисел, содержащий ненулевую часть верхнетреугольной матрицы (по строкам). Создать новый файл, содержащий все элементы данной матрицы (по строкам).

File87. Дан файл вещественных чисел, содержащий ненулевую часть нижнетреугольной матрицы (по строкам). Создать новый файл, содержащий все элементы данной матрицы (по строкам).

File88. Дан файл вещественных чисел, содержащий ненулевую часть трехдиагональной матрицы (по строкам). Создать новый файл, содержащий все элементы данной матрицы (по строкам).

File89. Даны два файла вещественных чисел с именами S_A и S_B, содержащие ненулевые части верхнетреугольных матриц A и B (по строкам). Создать новый файл с именем S_C, содержащий ненулевую часть произ-

ведения $A{\cdot}B$ (по строкам). Если матрицы A и B нельзя перемножать, то оставить файл S_C пустым.

File90. Даны два файла вещественных чисел с именами S_A и S_B, содержащие ненулевые части нижнетреугольных матриц A и B (по строкам). Создать новый файл с именем S_C, содержащий ненулевую часть произведения $A{\cdot}B$ (по строкам). Если матрицы A и B нельзя перемножать, то оставить файл S_C пустым.

Текстовые файлы

Условие вида «дан текстовый файл» означает, что в наборе исходных данных указано *имя* данного файла (текстовая строка). Все исходные файлы в заданиях данной группы считаются существующими. Элементы исходных файлов следует считывать с помощью стандартных процедур используемого языка программирования.

Если в задании требуется создать новый файл, то имя создаваемого файла также входит в набор исходных данных (и, как правило, является последним элементом этого набора). Записывать элементы в создаваемые (или модифицируемые) файлы следует с помощью стандартных процедур используемого языка программирования.

Максимальный размер исходных файлов не устанавливается, поэтому при решении заданий не следует использовать вспомогательные массивы, содержащие все элементы исходных файлов, однако допускается использование *вспомогательных файлов*.

Используемые в заданиях типизированные файлы удовлетворяют условиям, которые перечислены в начале раздела «Типизированные файлы».

Основные операции с текстовыми файлами

Text1°. Дано имя файла и целые положительные числа N и K. Создать текстовый файл с указанным именем и записать в него N строк, каждая из которых состоит из K символов «*» (звездочка).

Text2. Дано имя файла и целое число N ($0 < N < 27$). Создать текстовый файл с указанным именем и записать в него N строк: первая строка должна содержать *строчную* (то есть маленькую) латинскую букву «a», вторая

— буквы «ab», третья — буквы «abc» и т. д.; последняя строка должна содержать N начальных строчных латинских букв в алфавитном порядке.

Text3. Дано имя файла и целое число N ($0 < N < 27$). Создать текстовый файл с указанным именем и записать в него N строк длины N; строка с номером K ($K = 1, \ldots, N$) должна содержать K начальных *прописных* (то есть заглавных) латинских букв, дополненных справа символами «*» (звездочка). Например, для $N = 4$ файл должен содержать строки «A***», «AB**», «ABC*», «ABCD».

Text4°. Дан текстовый файл. Вывести количество содержащихся в нем символов и строк (маркеры концов строк EOLN и конца файла EOF при подсчете количества символов не учитывать).

Text5. Дана строка S и текстовый файл. Добавить строку S в конец файла.

Text6. Даны два текстовых файла. Добавить в конец первого файла содержимое второго файла.

Text7. Дана строка S и текстовый файл. Добавить строку S в начало файла.

Text8. Даны два текстовых файла. Добавить в начало первого файла содержимое второго файла.

Text9. Дано целое число K и текстовый файл. Вставить пустую строку перед строкой файла с номером K. Если строки с таким номером нет, то оставить файл без изменений.

Text10. Дано целое число K и текстовый файл. Вставить пустую строку после строки файла с номером K. Если строки с таким номером нет, то оставить файл без изменений.

Text11. Дан текстовый файл. Продублировать в нем все пустые строки.

Text12. Дана строка S и текстовый файл. Заменить в файле все пустые строки на строку S.

Text13. Дан непустой текстовый файл. Удалить из него первую строку.

Text14. Дан непустой текстовый файл. Удалить из него последнюю строку.

Text15. Дано целое число K и текстовый файл. Удалить из файла строку с номером K. Если строки с таким номером нет, то оставить файл без изменений.

Text16°. Дан текстовый файл. Удалить из него все пустые строки.

Text17. Даны два текстовых файла. Добавить в конец каждой строки первого файла соответствующую строку второго файла. Если второй файл короче первого, то оставшиеся строки первого файла не изменять.

Text18. Дано целое число K и текстовый файл. Удалить из каждой строки

файла первые K символов (если длина строки меньше K, то удалить из нее все символы).

Text19. Дан текстовый файл. Заменить в нем все прописные русские буквы на строчные, а все строчные — на прописные.

Text20. Дан текстовый файл. Заменить в нем все подряд идущие пробелы на один пробел.

Text21°. Дан текстовый файл, содержащий более трех строк. Удалить из него последние три строки.

Text22. Дано целое число K ($0 < K < 10$) и текстовый файл, содержащий более K строк. Удалить из файла последние K строк.

Text23. Дано целое число K ($0 < K < 10$) и текстовый файл, содержащий более K строк. Создать новый текстовый файл, содержащий K последних строк исходного файла.

Анализ и форматирование текста

Text24°. Дан текстовый файл. Найти количество абзацев в тексте, если абзацы отделяются друг от друга одной или несколькими пустыми строками.

Text25. Дано целое число K и текстовый файл. Удалить из файла абзац с номером K (абзацы отделяются друг от друга одной или несколькими пустыми строками). Пустые строки, предшествующие и следующие за удаляемым абзацем, не удалять. Если абзац с данным номером отсутствует, то оставить файл без изменений.

Text26. Дан текстовый файл. Найти количество абзацев в тексте, если первая строка каждого абзаца начинается с 5 пробелов («*красная строка*»). Пустые строки между абзацами не учитывать.

Text27. Дано целое число K и текстовый файл. Удалить из файла абзац с номером K (абзацы выделяются с помощью *красной строки* — см. задание Text26). Пустые строки между абзацами не учитывать и не удалять. Если абзац с данным номером отсутствует, то оставить файл без изменений.

Text28. Дан текстовый файл. Абзацы выделяются в нем с помощью *красной строки* (см. задание Text26), а пустых строк нет. Вставить между соседними абзацами по одной пустой строке (в начало и конец файла пустые строки не добавлять).

Text29. Дан текстовый файл. Вывести первое слово текста наибольшей длины. *Словом* считать набор символов, не содержащий пробелов и ограниченный пробелами или началом/концом строки.

Text30. Дан текстовый файл. Вывести последнее слово текста наименьшей длины. *Словом* считать набор символов, не содержащий пробелов и ограниченный пробелами или началом/концом строки.

Text31. Дано целое число K и текстовый файл. Создать строковый файл и записать в него все слова длины K из исходного файла. *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки. Если исходный файл не содержит слов длины K, то оставить результирующий файл пустым.

Text32. Дан символ C — *прописная* (заглавная) русская буква и текстовый файл. Создать строковый файл и записать в него все слова из исходного файла, начинающиеся на эту букву (прописную или строчную). *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки. Если исходный файл не содержит подходящих слов, то оставить результирующий файл пустым.

Text33. Дан символ C — *строчная* (маленькая) русская буква и текстовый файл. Создать строковый файл и записать в него все слова из исходного файла, содержащие хотя бы одну букву C (прописную или строчную). *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки. Если исходный файл не содержит подходящих слов, то оставить результирующий файл пустым.

Text34°. Дан текстовый файл, содержащий текст, выровненный по левому краю. Выровнять текст по правому краю, добавив в начало каждой непустой строки нужное количество пробелов (ширину текста считать равной 50).

Text35. Дан текстовый файл, содержащий текст, выровненный по левому краю. Выровнять текст по центру, добавив в начало каждой непустой строки нужное количество пробелов (ширину текста считать равной 50). Строки нечетной длины перед центрированием дополнять слева пробелом.

Text36. Дан текстовый файл, содержащий текст, выровненный по правому краю. Выровнять текст по центру, удалив из каждой непустой строки половину начальных пробелов. В строках с нечетным количеством начальных пробелов перед центрированием удалять первый начальный пробел.

Text37. Дан текстовый файл, содержащий текст, выровненный по левому краю.

Абзацы текста разделяются одной пустой строкой. Выровнять текст *по ширине* (то есть и по левому, и по правому краю), увеличив в каждой непустой строке (кроме последних строк абзацев) количество пробелов между словами, начиная с последнего пробела в строке (ширину текста считать равной 50).

Text38°. Дано целое число K (>25) и текстовый файл, содержащий текст, выровненный по левому краю. Абзацы текста отделяются друг от друга одной пустой строкой. Отформатировать текст так, чтобы его ширина не превосходила K позиций, и выровнять текст по левому краю, сохранив деление на абзацы. Пробелы в конце строк удалить. Сохранить отформатированный текст в новом текстовом файле.

Text39. Дано целое число K (>25) и текстовый файл, содержащий текст, выровненный по левому краю. Абзацы выделяются в нем с помощью *красной строки* (5 начальных пробелов), а пустых строк нет. Отформатировать текст так, чтобы его ширина не превосходила K позиций, и выровнять текст по левому краю, сохранив деление на абзацы. Пробелы в конце строк удалить. Сохранить отформатированный текст в новом текстовом файле.

Текстовые файлы с числовой информацией

В заданиях на обработку текстовых файлов с числовой информацией предполагается, что в изображениях вещественных чисел, содержащихся в текстовых файлах, дробная часть отделяется от целой части десятичной *точкой*.

Text40. Даны два файла целых чисел одинакового размера. Создать текстовый файл, содержащий эти числа, расположенные в два столбца шириной по 30 символов (в первом столбце содержатся числа из первого исходного файла, во втором — из второго файла). В начало и конец каждой строки текстового файла добавить разделитель «|» (код 124). Числа выравниваются по правому краю столбца.

Text41. Даны три файла целых чисел одинакового размера. Создать текстовый файл, содержащий эти числа, расположенные в три столбца шириной по 20 символов (в каждом столбце содержатся числа из соответствующего исходного файла). В начало и конец каждой строки текстового файла добавить разделитель «|» (код 124). Числа выравниваются по левому краю

столбца.

Text42°. Даны вещественные числа A, B и целое число N. Создать текстовый файл, содержащий таблицу значений функции $\sqrt{x}$ на промежутке $[A, B]$ с шагом $(B - A)/N$. Таблица состоит из двух столбцов: с аргументами x (10 позиций, из них 4 под дробную часть) и со значениями $\sqrt{x}$ (15 позиций, из них 8 под дробную часть). Столбцы выравниваются по правому краю.

Text43. Даны вещественные числа A, B и целое число N. Создать текстовый файл, содержащий таблицу значений функций $\sin(x)$ и $\cos(x)$ на промежутке $[A, B]$ с шагом $(B - A)/N$. Таблица состоит из трех столбцов: с аргументами x (8 позиций, из них 4 под дробную часть) и со значениями $\sin(x)$ и $\cos(x)$ (по 12 позиций, из них 8 под дробную часть). Столбцы выравниваются по правому краю.

Text44°. Дан текстовый файл, каждая строка которого изображает целое число, дополненное слева и справа несколькими пробелами. Вывести количество этих чисел и их сумму.

Text45. Дан текстовый файл, каждая строка которого изображает целое или вещественное число, дополненное слева и справа несколькими пробелами (вещественные числа имеют ненулевую дробную часть). Вывести количество чисел с ненулевой дробной частью и их сумму.

Text46. Дан текстовый файл, каждая строка которого содержит изображения нескольких чисел, разделенные пробелами (вещественные числа имеют ненулевую дробную часть). Создать файл вещественных чисел, содержащий (в том же порядке) все числа из исходного файла, имеющие ненулевую дробную часть.

Text47. Дан текстовый файл, каждая строка которого изображает целое или вещественное число, дополненное слева и справа несколькими пробелами (вещественные числа имеют ненулевую дробную часть). Вывести количество целых чисел и их сумму.

Text48. Дан текстовый файл, каждая строка которого содержит изображения нескольких чисел, разделенные пробелами (вещественные числа имеют ненулевую дробную часть). Создать файл целых чисел, содержащий все целые числа из исходного файла (в том же порядке).

Text49. Дан текстовый файл и файл целых чисел. Добавить в конец каждой строки текстового файла изображение соответствующего числа из файла целых чисел. Если файл целых чисел короче текстового файла, то оставшиеся строки текстового файла не изменять.

Text50. Дан текстовый файл. В каждой его строке первые 30 позиций отводятся под текст, а оставшаяся часть — под вещественное число. Создать два файла: строковый файл, содержащий текстовую часть исходного файла, и файл вещественных чисел, содержащий числа из исходного файла (в том же порядке).

Text51. Дан текстовый файл, содержащий таблицу из трех столбцов вещественных чисел. Ширина столбцов таблицы и способ их выравнивания являются произвольными, специальных символов-разделителей таблица не содержит. Создать три файла вещественных чисел, каждый из которых содержит числа из соответствующего столбца таблицы (в том же порядке).

Text52. Дан текстовый файл, содержащий таблицу из трех столбцов целых чисел. В начале и в конце каждой строки таблицы, а также между ее столбцами располагается *символ-разделитель*. Ширина столбцов таблицы, способ их выравнивания и вид символа-разделителя являются произвольными. Создать файл целых чисел, содержащий сумму чисел из каждой строки исходной таблицы.

Дополнительные задания на обработку текстовых файлов

Text53. Дан текстовый файл. Создать символьный файл, содержащий все знаки препинания, встретившиеся в текстовом файле (в том же порядке).

Text54. Дан текстовый файл. Создать символьный файл, содержащий все символы, встретившиеся в тексте, включая пробел и знаки препинания (без повторений). Символы располагать в порядке их первого появления в тексте.

Text55. Дан текстовый файл. Создать символьный файл, содержащий все символы, встретившиеся в тексте, включая пробел и знаки препинания (без повторений). Символы располагать в порядке возрастания их кодов.

Text56. Дан текстовый файл. Создать символьный файл, содержащий все символы, встретившиеся в тексте, включая пробел и знаки препинания (без повторений). Символы располагать в порядке убывания их кодов.

Text57°. Дан текстовый файл. Подсчитать число появлений в нем каждой *строчной* (то есть маленькой) русской буквы и создать строковый файл, элементы которого имеют вид «<буква>–<число ее появлений>» (например, «а–25»). Буквы, отсутствующие в тексте, в файл не включать. Строки упорядочить по возрастанию кодов букв.

Text58. Дан текстовый файл. Подсчитать число появлений в нем каждой *строчной* (то есть маленькой) русской буквы и создать строковый файл, элементы которого имеют вид «<буква>–<число ее появлений>» (например, «а–25»). Буквы, отсутствующие в тексте, в файл не включать. Строки упорядочить по убыванию числа появлений букв, а при равном числе появлений — по возрастанию кодов букв.

Text59. Дана строка S, состоящая из 10 цифр, и файл с русским текстом. Зашифровать файл, выполнив циклическую замену каждой русской буквы, стоящей на K-й позиции строки, на букву того же регистра, расположенную в алфавите на S_K-м месте после шифруемой буквы (для $K = 11$ снова используется смещение S_1 и т. д.). Букву «ё» в алфавите не учитывать, знаки препинания и пробелы не изменять.

Text60. Дана строка и файл с русским текстом, зашифрованным по правилу, описанному в задании Text59. Данная строка представляет собой первую расшифрованную строку текста. Расшифровать остальные строки и заменить в файле зашифрованный текст на расшифрованный. Если информации для расшифровки недостаточно, то исходный файл не изменять.

Составные типы данных в процедурах и функциях

В каждом задании данного раздела требуется описать процедуру или функцию и затем использовать ее для обработки исходных данных. Все параметры любой *функции* считаются входными. Для *процедур* всегда указывается, какие параметры являются выходными (или одновременно входными и выходными); если о виде параметра процедуры ничего не сказано, то он считается входным.

Одномерные и двумерные массивы

При вводе исходного массива вначале следует ввести его размер (одно число для одномерных массивов, два числа — количество строк и столбцов — для двумерных массивов-матриц), а затем — все его элементы.

Если в задании явно не указывается размер одномерного массива, являющегося параметром процедуры или функции, то предполагается, что этот размер может изменяться в пределах от 1 до 10. Для двумерных массивов-матриц предполагается, что число их строк и столбцов может меняться от 1

до 10. Индексы начальных элементов как одномерных, так и двумерных массивов всегда считаются равными 1.

При описании процедур, выполняющих преобразование массива, не следует использовать вспомогательный массив того же размера.

Param1°. Описать функцию MinElem(*A*, *N*) целого типа, находящую минимальный элемент целочисленного массива *A* размера *N*. С помощью этой функции найти минимальные элементы массивов *A*, *B*, *C* размера N_A, N_B, N_C соответственно.

Param2. Описать функцию MaxNum(*A*, *N*) целого типа, находящую номер максимального элемента вещественного массива *A* размера *N*. С помощью этой функции найти номера максимальных элементов массивов *A*, *B*, *C* размера N_A, N_B, N_C соответственно.

Param3. Описать процедуру MinmaxNum(*A*, *N*, *NMin*, *NMax*), находящую номера минимального и максимального элемента вещественного массива *A* размера *N*. Выходные параметры целого типа: *NMin* (номер минимального элемента) и *NMax* (номер максимального элемента). С помощью этой процедуры найти номера минимальных и максимальных элементов массивов *A*, *B*, *C* размера N_A, N_B, N_C соответственно.

Param4. Описать процедуру Invert(*A*, *N*), меняющую порядок следования элементов вещественного массива *A* размера *N* на противоположный (*инвертирование* массива). Массив *A* является входным и выходным параметром. С помощью этой процедуры инвертировать массивы *A*, *B*, *C* размера N_A, N_B, N_C соответственно.

Param5. Описать процедуру Smooth1(*A*, *N*), выполняющую *сглаживание* вещественного массива *A* размера *N* следующим образом: элемент A_K заменяется на среднее арифметическое первых *K* исходных элементов массива *A*. Массив *A* является входным и выходным параметром. С помощью этой процедуры выполнить пятикратное сглаживание данного массива *A* размера *N*, выводя результаты каждого сглаживания.

Param6. Описать процедуру Smooth2(*A*, *N*), выполняющую *сглаживание* вещественного массива *A* размера *N* следующим образом: элемент A_1 не изменяется, элемент A_K ($K = 2, \ldots, N$) заменяется на полусумму исходных элементов A_{K-1} и A_K. Массив *A* является входным и выходным параметром. С помощью этой процедуры выполнить пятикратное сглаживание данного массива *A* размера *N*, выводя результаты каждого сглаживания.

Param7. Описать процедуру Smooth3(A, N), выполняющую *сглаживание* вещественного массива A размера N следующим образом: каждый элемент массива заменяется на его среднее арифметическое с соседними элементами (при вычислении среднего арифметического используются *исходные* значения соседних элементов). Массив A является входным и выходным параметром. С помощью этой процедуры выполнить пятикратное сглаживание данного массива A размера N, выводя результаты каждого сглаживания.

Param8. Описать процедуру RemoveX(A, N, X), удаляющую из целочисленного массива A размера N элементы, равные целому числу X. Массив A и число N являются входными и выходными параметрами. С помощью этой процедуры удалить числа X_A, X_B, X_C из массивов A, B, C размера N_A, N_B, N_C соответственно и вывести размер и содержимое полученных массивов.

Param9. Описать процедуру RemoveForInc(A, N), удаляющую из вещественного массива A размера N «лишние» элементы так, чтобы оставшиеся элементы оказались упорядоченными по возрастанию: первый элемент не удаляется, второй элемент удаляется, если он меньше первого, третий — если он меньше предыдущего элемента, оставленного в массиве, и т. д. Например, массив 5.5, 2.5, 4.6, 7.2, 5.8, 9.4 должен быть преобразован к виду 5.5, 7.2, 9.4. Массив A и число N являются входными и выходными параметрами. С помощью этой процедуры преобразовать массивы A, B, C размера N_A, N_B, N_C соответственно и вывести размер и содержимое полученных массивов.

Param10. Описать процедуру DoubleX(A, N, X), дублирующую в целочисленном массиве A размера N элементы, равные целому числу X. Массив A и число N являются входными и выходными параметрами. С помощью этой процедуры продублировать числа X_A, X_B, X_C в массивах A, B, C размера N_A, N_B, N_C соответственно и вывести размер и содержимое полученных массивов.

Param11. Описать процедуру SortArray(A, N), выполняющую сортировку по возрастанию вещественного массива A размера N. Массив A является входным и выходным параметром. С помощью этой процедуры отсортировать массивы A, B, C размера N_A, N_B, N_C соответственно.

Param12. Описать процедуру SortIndex(A, N, I), формирующую для вещественного массива A размера N *индексный массив* I — массив целых чисел

того же размера, содержащий номера элементов массива A в том порядке, который соответствует возрастанию элементов массива A (сам массив A при этом не изменяется). Индексный массив I является выходным параметром. С помощью этой процедуры создать индексные массивы для массивов A, B, C размера N_A, N_B, N_C соответственно.

Param13. Описать процедуру Hill(A, N), меняющую порядок элементов вещественного массива A размера N на следующий: наименьший элемент массива располагается на первом месте, наименьший из оставшихся элементов — на последнем, следующий по величине располагается на втором месте, следующий — на предпоследнем и т. д. (в результате график значений элементов будет напоминать *холм*). Массив A является входным и выходным параметром. С помощью этой процедуры преобразовать массивы A, B, C размера N_A, N_B, N_C соответственно.

Param14. Описать процедуру Split1(A, N_A, B, N_B, C, N_C), формирующую по вещественному массиву A размера N_A два вещественных массива B и C размера N_B и N_C соответственно; при этом массив B содержит все элементы массива A с нечетными порядковыми номерами (1, 3, ...), а массив C — все элементы массива A с четными номерами (2, 4, ...). Массивы B и C и числа N_B и N_C являются выходными параметрами. Применить эту процедуру к данному массиву A размера N_A и вывести размер и содержимое полученных массивов B и C.

Param15. Описать процедуру Split2(A, N_A, B, N_B, C, N_C), формирующую по целочисленному массиву A размера N_A два целочисленных массива B и C размера N_B и N_C соответственно; при этом массив B содержит все четные числа из массива A, а массив C — все нечетные числа (в том же порядке). Массивы B и C и числа N_B и N_C являются выходными параметрами. Применить эту процедуру к данному массиву A размера N_A и вывести размер и содержимое полученных массивов B и C.

Param16. Описать процедуру ArrayToMatrRow(A, K, M, N, B), формирующую по вещественному массиву A размера K матрицу B размера $M \times N$ (матрица заполняется элементами массива A по строкам). «Лишние» элементы массива игнорируются; если элементов массива недостаточно, то оставшиеся элементы матрицы полагаются равными 0. Двумерный массив B является выходным параметром. С помощью этой процедуры на основе данного массива A размера K и целых чисел M и N сформировать матрицу B размера $M \times N$.

Param17°. Описать процедуру ArrayToMatrCol(A, K, M, N, B), формирующую по вещественному массиву A размера K матрицу B размера $M \times N$ (матрица заполняется элементами массива A по столбцам). «Лишние» элементы массива игнорируются; если элементов массива недостаточно, то оставшиеся элементы матрицы полагаются равными 0. Двумерный массив B является выходным параметром. С помощью этой процедуры на основе данного массива A размера K и целых чисел M и N сформировать матрицу B размера $M \times N$.

Param18. Описать процедуру Chessboard(M, N, A), формирующую по целым положительным числам M и N матрицу A размера $M \times N$, которая содержит числа 0 и 1, расположенные в «шахматном» порядке, причем $A_{1,1} = 0$. Двумерный целочисленный массив A является выходным параметром. С помощью этой процедуры по данным целым числам M и N сформировать матрицу A размера $M \times N$.

Param19. Описать функцию Norm1(A, M, N) вещественного типа, вычисляющую *норму* вещественной матрицы A размера $M \times N$:

$$\text{Norm1}(A, M, N) = \max \{|A_{1,J}| + |A_{2,J}| + \ldots + |A_{M,J}|\},$$

где максимум берется по всем J от 1 до N. Для данной матрицы A размера $M \times N$ найти Norm1(A, K, N), $K = 1, \ldots, M$.

Param20. Описать функцию Norm2(A, M, N) вещественного типа, вычисляющую *норму* вещественной матрицы A размера $M \times N$:

$$\text{Norm2}(A, M, N) = \max \{|A_{I,1}| + |A_{I,2}| + \ldots + |A_{I,N}|\},$$

где максимум берется по всем I от 1 до M. Для данной матрицы A размера $M \times N$ найти Norm2(A, K, N), $K = 1, \ldots, M$.

Param21. Описать функцию SumRow(A, M, N, K) вещественного типа, вычисляющую сумму элементов вещественной матрицы A размера $M \times N$, расположенных в K-й строке (если $K > M$, то функция возвращает 0). Для данной матрицы A размера $M \times N$ и трех данных K найти SumRow(A, M, N, K).

Param22. Описать функцию SumCol(A, M, N, K) вещественного типа, вычисляющую сумму элементов вещественной матрицы A размера $M \times N$, расположенных в K-м столбце (если $K > N$, то функция возвращает 0). Для данной матрицы A размера $M \times N$ и трех данных K найти SumCol(A, M, N, K).

Param23. Описать процедуру SwapRow(A, M, N, K_1, K_2), осуществляющую перемену местами строк вещественной матрицы A размера $M \times N$ с

номерами K_1 и K_2. Матрица A является входным и выходным параметром; если K_1 или K_2 больше M, то матрица не изменяется. Используя эту процедуру, поменять для данной матрицы A размера $M \times N$ строки с данными номерами K_1 и K_2.

Param24. Описать процедуру SwapCol(A, M, N, K_1, K_2), осуществляющую перемену местами столбцов вещественной матрицы A размера $M \times N$ с номерами K_1 и K_2. Матрица A является входным и выходным параметром; если K_1 или K_2 больше N, то матрица не изменяется. Используя эту процедуру, поменять для данной матрицы A размера $M \times N$ столбцы с данными номерами K_1 и K_2.

Param25. Описать процедуру Transp(A, M), выполняющую *транспонирование* (то есть зеркальное отражение относительно главной диагонали) квадратной вещественной матрицы A порядка M. Матрица A является входным и выходным параметром. Используя эту процедуру, транспонировать данную матрицу A порядка M.

Param26. Описать процедуру RemoveRows(A, M, N, K_1, K_2), удаляющую из вещественной матрицы A размера $M \times N$ строки с номерами от K_1 до K_2 включительно (предполагается, что $1 < K_1 \le K_2$). Если $K_1 > M$, то матрица не изменяется; если $K_2 > M$, то удаляются строки матрицы с номерами от K_1 до M. Двумерный массив A и числа M, N являются входными и выходными параметрами. Используя процедуру RemoveRows, удалить из данной матрицы A размера $M \times N$ строки с номерами от K_1 до K_2 и вывести размер полученной матрицы и ее элементы.

Param27. Описать процедуру RemoveCols(A, M, N, K_1, K_2), удаляющую из вещественной матрицы A размера $M \times N$ столбцы с номерами от K_1 до K_2 включительно (предполагается, что $1 < K_1 \le K_2$). Если $K_1 > N$, то матрица не изменяется; если $K_2 > N$, то удаляются столбцы матрицы с номерами от K_1 до N. Двумерный массив A и числа M, N являются входными и выходными параметрами. Используя процедуру RemoveCols, удалить из данной матрицы A размера $M \times N$ столбцы с номерами от K_1 до K_2 и вывести размер полученной матрицы и ее элементы.

Param28. Описать процедуру RemoveRowCol(A, M, N, K, L), удаляющую из вещественной матрицы A размера $M \times N$ строку и столбец, которые содержат элемент $A_{K,L}$ (предполагается, что $M > 1$ и $N > 1$; если $K > M$ или $L > N$, то матрица не изменяется). Двумерный массив A и числа M, N являются входными и выходными параметрами. Дана матрица A размера

$M \times N$ и числа K, L. Применить к матрице A процедуру RemoveRowCol и вывести размер полученной матрицы и ее элементы.

Param29. Описать процедуру SortCols(A, M, N), выполняющую сортировку по возрастанию столбцов целочисленной матрицы A размера $M \times N$ (столбцы сравниваются *лексикографически*: если первые элементы столбцов различны, то меньшим считается столбец, содержащий меньший первый элемент; если первые элементы столбцов равны, то анализируются их вторые элементы и т. д.). Двумерный массив A является входным и выходным параметром. Используя процедуру SortCols, отсортировать столбцы данной матрицы A размера $M \times N$.

Строки

Param30°. Описать функцию IsIdent(S) целого типа, проверяющую, является ли строка S допустимым *идентификатором*, то есть непустой строкой, которая содержит только латинские буквы, цифры и символ подчеркивания «» и не начинается с цифры. Если S является допустимым идентификатором, то функция возвращает 0. Если S является пустой строкой, то возвращается -1, если S начинается с цифры, то возвращается -2. Если S содержит недопустимые символы, то возвращается номер первого недопустимого символа. Проверить с помощью функции IsIdent пять данных строк.

Param31. Описать функцию FillStr(S, N) строкового типа, возвращающую строку длины N, заполненную повторяющимися копиями строки-*шаблона* S (последняя копия строки-шаблона может входить в результирующую строку частично). Используя эту функцию, сформировать по данному числу N и пяти данным строкам-шаблонам пять результирующих строк длины N.

Param32. Описать процедуру UpCaseRus(S), преобразующую все строчные русские буквы строки S в прописные (остальные символы строки S не изменяются). Строка S является входным и выходным параметром. Используя процедуру UpCaseRus, преобразовать пять данных строк.

Param33. Описать процедуру LowCaseRus(S), преобразующую все прописные русские буквы строки S в строчные (остальные символы строки S не изменяются). Строка S является входным и выходным параметром. Используя процедуру LowCaseRus, преобразовать пять данных строк.

Param34. Описать процедуру TrimLeftC(*S*, *C*), удаляющую в строке *S* начальные символы, совпадающие с символом *C*. Строка *S* является входным и выходным параметром. Дан символ *C* и пять строк. Используя процедуру TrimLeftC, преобразовать данные строки.

Param35. Описать процедуру TrimRightC(*S*, *C*), удаляющую в строке *S* конечные символы, совпадающие с символом *C*. Строка *S* является входным и выходным параметром. Дан символ *C* и пять строк. Используя процедуру TrimRightC, преобразовать данные строки.

Param36. Описать функцию InvertStr(*S*, *K*, *N*) строкового типа, возвращающую *инвертированную подстроку* строки *S*, содержащую в обратном порядке *N* символов строки *S*, начиная с ее *K*-го символа. Если *K* превосходит длину строки *S*, то возвращается пустая строка; если длина строки меньше *K* + *N*, то инвертируются все символы строки, начиная с ее *K*-го символа. Вывести значения функции InvertStr для данной строки *S* и каждой из трех пар положительных целых чисел: (K_1, N_1), (K_2, N_2), (K_3, N_3).

Param37. Описать функцию PosSub(S_0, *S*, *K*, *N*) целого типа, возвращающую номер позиции, начиная с которой в строке *S* содержится первое вхождение строки S_0, причем анализируются только *N* символов строки *S*, начиная с ее *K*-го символа (таким образом, PosSub обеспечивает *поиск в подстроке*). Если *K* превосходит длину строки *S*, то возвращается 0, если длина строки меньше *K* + *N*, то анализируются все символы строки, начиная с ее *K*-го символа. Если в требуемой подстроке строки *S* вхождения S_0 отсутствуют, то функция возвращает 0. Вывести значения функции PosSub для данных строк S_0, *S* и каждой из трех пар положительных целых чисел: (K_1, N_1), (K_2, N_2), (K_3, N_3).

Param38. Описать функцию PosLast(S_0, *S*) целого типа, возвращающую номер позиции, начиная с которой в строке *S* содержится последнее вхождение подстроки S_0. Если в строке *S* отсутствуют подстроки S_0, то функция возвращает 0. Вывести значения этой функции для пяти данных пар строк S_0 и *S*.

Param39. Описать функцию PosK(S_0, *S*, *K*) целого типа, возвращающую номер позиции, начиная с которой в строке *S* содержится *K*-е вхождение подстроки S_0 (*K* > 0). Если количество вхождений S_0 в строке *S* меньше *K*, то функция возвращает 0. Считать, что перекрывающихся вхождений подстрок S_0 строка *S* не содержит. Вывести значения этой функции для пяти данных троек: S_0, *S* и *K*.

Param40°. Описать функцию WordK(*S*, *K*) строкового типа, возвращающую *K*-е слово строки *S* (*словом* считается набор символов, не содержащий пробелов и ограниченный пробелами или началом/концом строки). Если количество слов в строке меньше *K*, то функция возвращает пустую строку. Используя эту функцию, выделить из данной строки *S* слова с данными номерами K_1, K_2, K_3.

Param41. Описать процедуру SplitStr(*S*, *W*, *N*), которая формирует по данной строке *S* массив *W* слов, входящих в *S* (массив *W* и его размер *N* являются выходными параметрами). *Словом* считается набор символов, не содержащий пробелов и ограниченный пробелами или началом/концом строки; предполагается, что строка *S* содержит не более 10 слов. Используя функцию SplitStr, найти количество слов *N*, содержащихся в данной строке *S*, и сами эти слова.

Param42. Описать функцию CompressStr(*S*) строкового типа, выполняющую *сжатие* строки *S* по следующему правилу: каждая подстрока строки *S*, состоящая из более чем четырех одинаковых символов *C*, заменяется текстом вида «*C*{*K*}», где *K* — количество символов *C* (предполагается, что строка *S* не содержит фигурных скобок «{» и «}»). Например, для строки *S* = «bbbccccce» функция вернет строку «bbbc{5}e». С помощью функции CompressStr сжать пять данных строк.

Param43. Описать функцию DecompressStr(*S*) строкового типа, восстанавливающую строку, сжатую функцией CompressStr (см. задание Param42). Параметр *S* содержит сжатую строку; восстановленная строка является возвращаемым значением функции. С помощью функции DecompressStr восстановить пять данных сжатых строк.

Param44. Описать функцию DecToBin(*N*) строкового типа, возвращающую строковое представление целого неотрицательного числа *N* в двоичной системе счисления. Результирующая строка состоит из символов «0»–«1» и не содержит ведущих нулей (за исключением представления числа 0). Используя эту функцию, получить двоичные представления пяти данных чисел.

Param45. Описать функцию DecToHex(*N*) строкового типа, возвращающую строковое представление целого неотрицательного числа *N* в 16-ричной системе счисления. Результирующая строка состоит из символов «0»–«9», «A»–«F» и не содержит ведущих нулей (за исключением представления числа 0). Используя эту функцию, получить 16-ричные представления

пяти данных чисел.

Param46. Описать функцию BinToDec(*S*) целого типа, определяющую целое неотрицательное число по его строковому представлению *S* в двоичной системе счисления. Параметр *S* имеет строковый тип, состоит из символов «0»–«1» и не содержит ведущих нулей (за исключением значения «0»). Используя эту функцию, вывести пять чисел, для которых даны их двоичные представления.

Param47. Описать функцию HexToDec(*S*) целого типа, определяющую целое неотрицательное число по его строковому представлению *S* в 16-ричной системе счисления. Параметр *S* имеет строковый тип, состоит из символов «0»–«9», «A»–«F» и не содержит ведущих нулей (за исключением значения «0»). Используя эту функцию, вывести пять чисел, для которых даны их 16-ричные представления.

Файлы

Param48. Описать функцию IntFileSize(*S*) целого типа, возвращающую количество элементов в файле целых чисел с именем *S*. Если файл не существует, то функция возвращает -1. С помощью этой функции найти количество элементов в трех файлах с данными именами.

Param49°. Описать функцию LineCount(*S*) целого типа, возвращающую количество строк в текстовом файле с именем *S*. Если файл не существует, то функция возвращает -1. С помощью этой функции найти количество строк в трех файлах с данными именами.

Param50. Описать процедуру InvertIntFile(*S*), меняющую порядок следования элементов файла целого типа с именем *S* на противоположный. Если файл не существует или содержит менее двух элементов, то процедура не выполняет никаких действий. Обработать с помощью этой процедуры три файла с данными именами.

Param51. Описать процедуру AddLineNumbers(*S*, *N*, *K*, *L*), добавляющую в начало каждой строки существующего текстового файла с именем *S* ее порядковый номер: первая строка получает номер *N*, вторая — $N+1$ и т. д. Номер отображается в *K* позициях, выравнивается по правому краю и отделяется от последующего текста *L* пробелами ($K > 0$, $L > 0$). Если строка файла является пустой, то она также нумеруется, но пробелы после номера не добавляются. Применить эту процедуру к данному файлу, используя указанные значения *N*, *K* и *L*.

Param52. Описать процедуру RemoveLineNumbers(S), удаляющую из начала каждой строки существующего текстового файла с именем S ее порядковый номер, а также пробелы, отделяющие номер от последующего текста (предполагается, что номера могли быть ранее добавлены к строкам файла с помощью процедуры AddLineNumbers — см. задание Param51). Если строки не содержат номеров, то процедура не выполняет никаких действий. Применить эту процедуру к файлу с данным именем.

Param53°. Описать процедуру SplitIntFile(S_0, K, S_1, S_2), копирующую первые K (≥ 0) элементов существующего файла целых чисел с именем S_0 в новый файл целых чисел с именем S_1, а остальные элементы — в новый файл целых чисел с именем S_2. Один из созданных файлов может остаться пустым. Применить эту процедуру к файлу с данным именем S_0, используя указанные значения K, S_1 и S_2.

Param54. Описать процедуру SplitText(S_0, K, S_1, S_2), копирующую первые K (≥ 0) строк существующего текстового файла с именем S_0 в новый текстовый файл с именем S_1, а остальные строки — в новый текстовый файл с именем S_2. Один из созданных файлов может остаться пустым. Применить эту процедуру к файлу с данным именем S_0, используя указанные значения K, S_1 и S_2.

Param55. Описать процедуру StringFileToText(S), преобразующую двоичный строковый файл с именем S в текстовый файл с тем же именем. Используя эту процедуру, преобразовать два данных строковых файла с именами S_1 и S_2 в текстовые.

Param56. Описать процедуру TextToStringFile(S), преобразующую текстовый файл с именем S в двоичный строковый файл с тем же именем. Используя эту процедуру, преобразовать два данных текстовых файла с именами S_1 и S_2 в строковые.

Param57. Описать процедуру EncodeText(S, K), которая шифрует текстовый файл с именем S, выполняя циклическую замену каждой русской буквы на букву того же регистра, расположенную в алфавите на K-й позиции после шифруемой буквы ($0 < K < 10$). Например, при $K = 3$ «А» перейдет в «Г», «я» — в «в». Букву «ё» в алфавите не учитывать, считая, что за буквой «е» сразу идет «ж». Символы, не являющиеся русскими буквами, при шифровании не изменять. Используя эту процедуру и зная кодовое смещение K, зашифровать файл с указанным именем.

Param58. Описать процедуру DecodeText(S, K), которая дешифрует текстовый

файл с именем *S*, зашифрованный с использованием кодового смещения *K* (способ шифрования описан в задании Param57). Используя эту процедуру и зная кодовое смещение *K*, расшифровать файл с указанным именем.

Записи

При вводе и выводе каждой даты в заданиях Param59–Param63 вначале указывается день, затем номер месяца, затем год. При вводе каждой точки в заданиях Param64–Param70 вначале указывается ее абсцисса (*x*-координата), затем ее ордината (*y*-координата).

Param59°. Описать тип TDate — запись с полями целого типа Day (день), Month (месяц) и Year (год) — и функцию LeapYear(*D*) логического типа с параметром типа TDate, которая возвращает TRUE, если год в дате *D* является високосным, и FALSE в противном случае. Вывести значение функции LeapYear для пяти данных дат (предполагается, что все даты являются правильными). *Високосным* считается год, делящийся на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400.

Param60°. Используя тип TDate и функцию LeapYear (см. задание Param59), описать функцию DaysInMonth(*D*) целого типа с параметром типа TDate, которая возвращает количество дней для месяца, указанного в дате *D*. Вывести значение функции DaysInMonth для пяти данных дат (предполагается, что все даты являются правильными).

Param61°. Используя тип TDate и функцию DaysInMonth (см. задания Param59 и Param60), описать функцию CheckDate(*D*) целого типа с параметром типа TDate, которая проверяет правильность даты, указанной в параметре *D*. Если дата *D* является правильной, то функция возвращает 0; если в дате указан неверный номер месяца, то функция возвращает 1; если в дате указан неверный день для допустимого месяца, то возвращается 2. Вывести значение функции CheckDate для пяти данных дат.

Param62. Используя тип TDate и функции DaysInMonth и CheckDate (см. задания Param59–Param61), описать процедуру PrevDate(*D*) с параметром типа TDate, которая преобразует дату *D* к предыдущей дате (если дата *D* является неправильной, то она не изменяется). Запись *D* является входным и выходным параметром. Применить процедуру PrevDate к пяти данным датам.

Param63. Используя тип TDate и функции DaysInMonth и CheckDate (см. задания Param59–Param61), описать процедуру NextDate(*D*) с параметром типа TDate, которая преобразует дату *D* к следующей дате (если дата *D* является неправильной, то она не изменяется). Запись *D* является входным и выходным параметром. Применить процедуру NextDate к пяти данным датам.

Param64. Описать тип TPoint — запись с полями вещественного типа *X* и *Y* (координаты точки на плоскости) — и функцию Leng(*A*, *B*) вещественного типа, находящую длину отрезка *AB* на плоскости по координатам его концов:

$$|AB| = \sqrt{(A.X - B.X)^2 + (A.Y - B.Y)^2}$$

(*A* и *B* — параметры типа TPoint). С помощью этой функции найти длины отрезков *AB*, *AC*, *AD*, если даны координаты точек *A*, *B*, *C*, *D*.

Param65. Используя тип TPoint и функцию Leng (см. задание Param64), описать тип TTriangle — запись с полями *A*, *B*, *C* типа TPoint (вершины треугольника) — и функцию Perim(*T*) вещественного типа, находящую периметр треугольника *T* (*T* — параметр типа TTriangle). С помощью этой функции найти периметры треугольников *ABC*, *ABD*, *ACD*, если даны координаты точек *A*, *B*, *C*, *D*.

Param66. Используя типы TPoint, TTriangle и функции Leng и Perim (см. задания Param64 и Param65), описать функцию Area(*T*) вещественного типа, находящую площадь треугольника *T* (*T* — параметр типа TTriangle) по *формуле Герона*:

$$S_{ABC} = \sqrt{p \cdot (p - |AB|) \cdot (p - |AC|) \cdot (p - |BC|)},$$

где *p* — *полупериметр*. С помощью этой функции найти площади треугольников *ABC*, *ABD*, *ACD*, если даны координаты точек *A*, *B*, *C*, *D*.

Param67. Используя типы TPoint, TTriangle и функции Leng и Area (см. задания Param64–Param66), описать функцию Dist(*P*, *A*, *B*) вещественного типа (*P*, *A*, *B* — параметры типа TPoint), находящую расстояние *D*(*P*, *AB*) от точки *P* до прямой *AB* по формуле

$$D(P, AB) = 2 \cdot S_{PAB} / |AB|,$$

где S_{PAB} — площадь треугольника *PAB*. С помощью этой функции найти расстояния от точки *P* до прямых *AB*, *AC*, *BC*, если даны координаты точек *P*, *A*, *B*, *C*.

Param68. Используя типы TPoint, TTriangle и функцию Dist (см. задания Param64, Param65, Param67), описать процедуру Heights(*T*, h_1, h_2, h_3),

находящую высоты h_1, h_2, h_3 треугольника T (T — входной параметр типа TTriangle, h_1, h_2, h_3 — выходные вещественные параметры), проведенные соответственно из вершин $T.A$, $T.B$, $T.C$. С помощью этой процедуры найти высоты треугольников ABC, ABD, ACD, если даны координаты точек A, B, C, D.

Param69. Используя тип TPoint и функцию Leng (см. задание Param64), описать функцию PerimN(P, N) вещественного типа, находящую периметр N-угольника, вершины которого (в порядке их обхода) передаются в массиве P размера N (>2) с элементами типа TPoint. С помощью этой функции найти периметры трех многоугольников, если дано число их сторон и координаты их вершин.

Param70. Используя типы TPoint, TTriangle и функцию Area (см. задания Param64–Param66), описать функцию AreaN(P, N) вещественного типа, находящую площадь выпуклого N-угольника, вершины которого (в порядке их обхода) передаются в массиве P размера N (>2) с элементами типа TPoint. С помощью этой функции найти площади трех многоугольников, если дано число их сторон и координаты их вершин.

Рекурсия

Простейшие рекурсивные алгоритмы

Задания этого раздела можно легко решить и *без использования рекурсии*. Данное обстоятельство связано с тем, что в заданиях рассматриваются *простейшие* примеры рекурсии, легко сводимые к итерационным алгоритмам. Более того, в некоторых случаях непосредственные вычисления по рекурсивным формулам оказываются весьма неэффективными (см., например, задания Recur4 и Recur6). Однако именно на подобных примерах проще всего получить первоначальные навыки разработки рекурсивных алгоритмов.

Recur1°. Описать рекурсивную функцию Fact(N) вещественного типа, вычисляющую значение *факториала*

$$N! = 1 \cdot 2 \cdot \ldots \cdot N$$

($N > 0$ — параметр целого типа). С помощью этой функции вычислить факториалы пяти данных чисел.

Recur2. Описать рекурсивную функцию Fact2(N) вещественного типа, вычисляющую значение *двойного факториала*

$$N!! = N\cdot(N-2)\cdot(N-4)\cdot\ldots$$

($N > 0$ — параметр целого типа; последний сомножитель в произведении равен 2, если N — четное число, и 1, если N — нечетное). С помощью этой функции вычислить двойные факториалы пяти данных чисел.

Recur3. Описать рекурсивную функцию PowerN(X, N) вещественного типа, находящую значение N-й степени числа X по формулам:

$$X^0 = 1,$$

$X^N = (X^{N/2})^2$ при *четных* $N > 0$, $\quad X^N = X\cdot X^{N-1}$ при *нечетных* $N > 0$,

$$X^N = 1/X^{-N} \text{ при } N < 0$$

($X \neq 0$ — вещественное число, N — целое; в формуле для четных N должна использоваться операция *целочисленного деления*). С помощью этой функции найти значения X^N для данного X при пяти данных значениях N.

Recur4°. Описать рекурсивную функцию Fib1(N) целого типа, вычисляющую N-й элемент последовательности *чисел Фибоначчи* (N — целое число):

$$F_1 = F_2 = 1, \qquad F_K = F_{K-2} + F_{K-1}, \quad K = 3, 4, \ldots .$$

С помощью этой функции найти пять чисел Фибоначчи с данными номерами, и вывести эти числа вместе с количеством рекурсивных вызовов функции Fib1, потребовавшихся для их нахождения.

Recur5°. Описать рекурсивную функцию Fib2(N) целого типа, вычисляющую N-й элемент последовательности *чисел Фибоначчи* (N — целое число):

$$F_1 = F_2 = 1, \qquad F_K = F_{K-2} + F_{K-1}, \quad K = 3, 4, \ldots .$$

Считать, что номер N не превосходит 20. Для уменьшения количества рекурсивных вызовов по сравнению с функцией Fib1 (см. задание Recur4) создать вспомогательный массив для хранения *уже вычисленных* чисел Фибоначчи и обращаться к нему при выполнении функции Fib2. С помощью функции Fib2 найти пять чисел Фибоначчи с данными номерами.

Recur6. Описать рекурсивную функцию Combin1(N, K) целого типа, находящую $C(N,K)$ — *число сочетаний* из N элементов по K — с помощью рекуррентного соотношения:

$$C(N,0) = C(N,N) = 1,$$

$$C(N,K) = C(N-1,K) + C(N-1,K-1) \quad \text{при } 0 < K < N.$$

Параметры функции — целые числа; $N > 0$, $0 \le K \le N$. Дано число N и пять различных значений K. Вывести числа $C(N,K)$ вместе с количеством рекурсивных вызовов функции Combin1, потребовавшихся для их нахождения.

Recur7. Описать рекурсивную функцию Combin2(N, K) целого типа, находящую $C(N,K)$ — *число сочетаний* из N элементов по K — с помощью рекуррентного соотношения:

$$C(N,0) = C(N,N) = 1,$$
$$C(N,K) = C(N-1,K) + C(N-1,K-1) \quad \text{при } 0 < K < N.$$

Параметры функции — целые числа; $N > 0$, $0 \le K \le N$. Считать, что параметр N не превосходит 20. Для уменьшения количества рекурсивных вызовов по сравнению с функцией Combin1 (см. задание Recur6) описать вспомогательный двумерный массив для хранения *уже вычисленных* чисел $C(N,K)$ и обращаться к нему при выполнении функции Combin2. С помощью функции Combin2 найти числа $C(N,K)$ для данного значения N и пяти различных значений K.

Recur8. Описать рекурсивную функцию RootK(X, K, N) вещественного типа, находящую приближенное значение корня K-й степени из числа X по формуле:

$$Y_0 = 1, \qquad Y_{N+1} = Y_N - (Y_N - X/(Y_N)^{K-1})/K,$$

где Y_N обозначает RootK(X, K, N) при фиксированных X и K. Параметры функции: X (>0) — вещественное число, K (>1) и N (>0) — целые. С помощью функции RootK найти для данного числа X приближенные значения его корня K-й степени при шести данных значениях N.

Recur9. Описать рекурсивную функцию NOD(A, B) целого типа, находящую *наибольший общий делитель* (НОД) двух целых положительных чисел A и B, используя *алгоритм Евклида*:

$$\text{НОД}(A,B) = \text{НОД}(B,\ A \bmod B), \quad \text{если } B \ne 0; \qquad \text{НОД}(A,0) = A.$$

С помощью этой функции найти НОД(A, B), НОД(A, C), НОД(A, D), если даны числа A, B, C, D.

Recur10°. Описать рекурсивную функцию DigitSum(K) целого типа, которая находит сумму цифр целого числа K, не используя оператор цикла. С помощью этой функции найти суммы цифр для пяти данных целых чисел.

Recur11. Описать рекурсивную функцию MaxElem(A, N) целого типа, которая находит максимальный элемент целочисленного массива A размера N ($1 \le N \le 10$), не используя оператор цикла. С помощью этой функции найти максимальные элементы массивов A, B, C размера N_A, N_B, N_C соответственно.

Recur12. Описать рекурсивную функцию DigitCount(S) целого типа, которая находит количество цифр в строке S, не используя оператор цикла. С

помощью этой функции найти количество цифр в каждой из пяти данных строк.

Recur13. Описать рекурсивную функцию Palindrom(*S*) логического типа, возвращающую TRUE, если строка *S* является *палиндромом* (то есть читается одинаково слева направо и справа налево), и FALSE в противном случае. Оператор цикла в теле функции не использовать. Вывести значения функции Palindrom для пяти данных строк.

Разбор выражений

Во всех заданиях данного пункта предполагается, что исходные строки, определяющие выражения, не содержат пробелов. При выполнении заданий не следует использовать оператор цикла.

Recur14°. Вывести значение целочисленного выражения, заданного в виде строки *S*. Выражение определяется следующим образом:

<выражение> ::= <цифра> | <выражение> + <цифра> |
<выражение> – <цифра>

Recur15°. Вывести значение целочисленного выражения, заданного в виде строки *S*. Выражение определяется следующим образом:

<выражение> ::= <терм> | <выражение> + <терм> |
<выражение> – <терм>
<терм> ::= <цифра> | <терм> * <цифра>

Recur16°. Вывести значение целочисленного выражения, заданного в виде строки *S*. Выражение определяется следующим образом:

<выражение> ::= <терм> | <выражение> + <терм> |
<выражение> – <терм>
<терм> ::= <элемент> | <терм> * <элемент>
<элемент> ::= <цифра> | (<выражение>)

Recur17°. Вывести значение целочисленного выражения, заданного в виде строки *S*. Выражение определяется следующим образом:

<выражение> ::= <цифра> |
(<выражение><знак><выражение>)
<знак> ::= + | – | *

Recur18°. Проверить правильность выражения, заданного в виде непустой строки *S* (выражение определяется по тем же правилам, что и в задании

Recur17). Если выражение составлено правильно, то вывести TRUE, иначе вывести FALSE.

Recur19. Проверить правильность выражения, заданного в виде непустой строки *S* (выражение определяется по тем же правилам, что и в задании Recur17). Если выражение составлено правильно, то вывести 0, в противном случае вывести номер первого ошибочного, лишнего или недостающего символа в строке *S*.

Recur20. Вывести значение целочисленного выражения, заданного в виде строки *S*. Выражение определяется следующим образом (функция M возвращает максимальный из своих параметров, а функция m — минимальный):

<выражение> ::= <цифра> | M(<выражение>,<выражение>) | m(<выражение>,<выражение>)

Recur21°. Вывести значение логического выражения, заданного в виде строки *S*. Выражение определяется следующим образом («T» — TRUE, «F» — FALSE):

<выражение> ::= T | F | And(<выражение>,<выражение>) | Or(<выражение>,<выражение>)

Recur22. Вывести значение целочисленного выражения, заданного в виде строки *S*. Выражение определяется следующим образом (функция M возвращает максимальный из своих параметров, а функция m — минимальный):

<выражение> ::= <цифра> | M(<параметры>) | m(<параметры>)
<параметры> ::= <выражение> | <выражение>,<параметры>

Recur23. Вывести значение логического выражения, заданного в виде строки *S*. Выражение определяется следующим образом («T» — TRUE, «F» — FALSE):

<выражение> ::= T | F | And(<параметры>) | Or(<параметры>)
<параметры> ::= <выражение> | <выражение>,<параметры>

Recur24. Вывести значение логического выражения, заданного в виде строки *S*. Выражение определяется следующим образом («T» — TRUE, «F» — FALSE):

<выражение> ::= T | F | And(<параметры>) | Or(<параметры>) | Not(<выражение>)
<параметры> ::= <выражение> | <выражение>,<параметры>

Перебор с возвратом

Recur25°. Дано дерево глубины N, каждая внутренняя вершина которого имеет K (<10) непосредственных потомков (нумеруются от 1 до K). Корень дерева имеет номер 0. Записать в текстовый файл с данным именем все возможные пути, ведущие от корня к листьям. Перебирать пути, начиная с «самого левого» и заканчивая «самым правым» (при этом первыми заменять конечные элементы пути).

Recur26. Дано дерево глубины N, каждая внутренняя вершина которого имеет K (<10) непосредственных потомков (нумеруются от 1 до K). Корень дерева имеет номер 0. Записать в текстовый файл с данным именем все пути, ведущие от корня к листьям и удовлетворяющие следующему условию: никакие соседние элементы пути не нумеруются одной и той же цифрой. Порядок перебора путей такой же, как в задании Recur25.

Recur27°. Дано дерево глубины N (N — четное), каждая внутренняя вершина которого имеет 2 непосредственных потомка: A с весом 1 и B с весом -1. Корень дерева C имеет вес 0. Записать в текстовый файл с данным именем все пути от корня к листьям, удовлетворяющие следующему условию: суммарный вес элементов пути равен 0. Порядок перебора путей такой же, как в задании Recur25.

Recur28. Дано дерево глубины N того же типа, что и в задании Recur27. Записать в текстовый файл с данным именем все пути от корня к листьям, удовлетворяющие следующему условию: суммарный вес элементов для любого начального отрезка пути неотрицателен. Порядок перебора путей такой же, как в задании Recur25.

Recur29. Дано дерево глубины N, каждая внутренняя вершина которого имеет 3 непосредственных потомка: A с весом 1, B с весом 0 и C с весом -1. Корень дерева D имеет вес 0. Записать в текстовый файл с данным именем все пути от корня к листьям, удовлетворяющие следующим условиям: суммарный вес элементов для любого начального отрезка пути неположителен, а суммарный вес всех элементов пути равен 0. Порядок перебора путей такой же, как в задании Recur25.

Recur30. Дано дерево глубины N того же типа, что и в задании Recur29. Записать в текстовый файл с данным именем все пути от корня к листьям, удовлетворяющие следующим условиям: никакие соседние элементы пути не обозначаются одной и той же буквой, а суммарный вес всех элементов пути равен 0. Порядок перебора путей такой же, как в задании

Recur25.

Динамические структуры данных

Данный раздел содержит описание варианта группы Dynamic, предназначенного для языков программирования Pascal и C++.

Все числа, упоминаемые в заданиях данной группы, являются *целыми*. Все указатели имеют тип PNode и указывают на записи типа TNode. Типы PNode и TNode описаны в варианте задачника для языка Pascal следующим образом:

```
type
   PNode = ^TNode;
   TNode = record
      Data: integer;
      Next: PNode;
      Prev: PNode;
   end;
```

Аналогично описываются эти типы в варианте задачника для языка C++:

```
struct TNode
{
   int Data;
   TNode* Next;
   TNode* Prev;
};

typedef TNode* PNode;
```

Во вводных заданиях Dynamic1–Dynamic2, а также в заданиях на стек и очередь (Dynamic3–Dynamic28) при работе с записями типа TNode используются только поля Data и Next (см. задание Dynamic1); в заданиях на списки (Dynamic29–Dynamic80) используются все поля записи TNode (см. задание Dynamic29).

Так как переменные типа «указатель» предназначены для хранения *адресов*, в формулировках заданий слова «указатель» (на элемент данных) и «адрес» (элемента данных) используются как синонимы.

В заданиях, в которых идет речь о номерах элементов списка, предполагается, что элементы списка нумеруются от 1.

Для обозначения нулевого указателя в формулировках заданий используется имя nil (как в языке Pascal).

Dynamic1. Дан адрес P_1 записи типа TNode, содержащей поле Data (целого типа) и поле Next (типа PNode — указателя на TNode). Эта запись связана полем Next со следующей записью того же типа. Вывести значения полей Data обеих записей, а также адрес P_2 следующей записи.

Dynamic2°. Дан адрес P_1 записи типа TNode. Эта запись связана полем Next со следующей записью того же типа, она, в свою очередь, — со следующей, и так далее до записи, поле Next которой равно nil (таким образом, возникает *цепочка* связанных записей). Вывести значения полей Data для всех элементов цепочки, *длину цепочки* (то есть число ее элементов) и адрес ее последнего элемента.

Стек

В заданиях Dynamic3–Dynamic13 структура *«стек»* (stack) моделируется цепочкой связанных узлов-записей типа TNode (см. задание Dynamic2). Поле Next последнего элемента цепочки равно nil. *Вершиной стека* (top) считается первый элемент цепочки. Для доступа к стеку используется указатель на его вершину (для пустого стека данный указатель полагается равным nil). *Значением* элемента стека считается значение его поля Data.

Dynamic3°. Дано число D и указатель P_1 на вершину непустого стека. Добавить элемент со значением D в стек и вывести адрес P_2 новой вершины стека.

Dynamic4. Дано число N (>0) и набор из N чисел. Создать стек, содержащий исходные числа (последнее число будет вершиной стека), и вывести указатель на его вершину.

Dynamic5°. Дан указатель P_1 на вершину непустого стека. Извлечь из стека первый (верхний) элемент и вывести его значение D, а также адрес P_2 новой вершины стека. Если после извлечения элемента стек окажется пустым, то положить P_2 = nil. После извлечения элемента из стека освободить память, занимаемую этим элементом.

Dynamic6. Дан указатель P_1 на вершину стека, содержащего не менее десяти элементов. Извлечь из стека первые девять элементов и вывести их

значения. Вывести также адрес новой вершины стека. После извлечения элементов из стека освобождать память, которую они занимали.

Dynamic7. Дан указатель P_1 на вершину стека (если стек пуст, то P_1 = nil). Извлечь из стека все элементы и вывести их значения. Вывести также количество извлеченных элементов N (для пустого стека вывести 0). После извлечения элементов из стека освобождать память, которую они занимали.

Dynamic8°. Даны указатели P_1 и P_2 на вершины двух непустых стеков. Переместить все элементы из первого стека во второй (в результате элементы первого стека будут располагаться во втором стеке в порядке, обратном исходному) и вывести адрес новой вершины второго стека. Операции выделения и освобождения памяти не использовать.

Dynamic9°. Даны указатели P_1 и P_2 на вершины двух непустых стеков. Перемещать элементы из первого стека во второй, пока значение вершины первого стека не станет четным (перемещенные элементы первого стека будут располагаться во втором стеке в порядке, обратном исходному). Если в первом стеке нет элементов с четными значениями, то переместить из первого стека во второй все элементы. Вывести адреса новых вершин первого и второго стека (если первый стек окажется пустым, то вывести для него константу nil). Операции выделения и освобождения памяти не использовать.

Dynamic10°. Дан указатель P_1 на вершину непустого стека. Создать два новых стека, переместив в первый из них все элементы исходного стека с четными значениями, а во второй — с нечетными (элементы в новых стеках будут располагаться в порядке, обратном исходному; один из этих стеков может оказаться пустым). Вывести адреса вершин полученных стеков (для пустого стека вывести nil). Операции выделения и освобождения памяти не использовать.

Dynamic11°. Дан указатель P_1 на вершину стека (если стек пуст, то P_1 = nil). Также дано число N (>0) и набор из N чисел. Описать тип TStack — запись с одним полем Top типа PNode (поле указывает на *вершину* стека) — и процедуру Push(S, D), которая добавляет в стек S новый элемент со значением D (S — входной и выходной параметр типа TStack, D — входной параметр целого типа). С помощью процедуры Push добавить в исходный стек данный набор чисел (последнее число будет вершиной стека) и вывести адрес новой вершины стека.

Dynamic12°. Дан указатель P_1 на вершину стека, содержащего не менее пяти элементов. Используя тип TStack (см. задание Dynamic11), описать функцию Pop(S) целого типа, которая извлекает из стека S первый (верхний) элемент, возвращает его значение и освобождает память, которую занимал извлеченный элемент (S — входной и выходной параметр типа TStack). С помощью функции Pop извлечь из исходного стека пять элементов и вывести их значения. Вывести также указатель на новую вершину стека (если результирующий стек окажется пустым, то этот указатель должен быть равен nil).

Dynamic13. Дан указатель P_1 на вершину стека. Используя тип TStack (см. задание Dynamic11), описать функции StackIsEmpty(S) логического типа (возвращает TRUE, если стек S пуст, и FALSE в противном случае) и Peek(S) целого типа (возвращает значение вершины непустого стека S, не удаляя ее из стека). В обеих функциях переменная S является входным параметром типа TStack. С помощью этих функций, а также функции Pop из задания Dynamic12, извлечь из исходного стека пять элементов (или все содержащиеся в нем элементы, если их менее пяти) и вывести их значения. Вывести также значение функции StackIsEmpty для результирующего стека и, если результирующий стек не является пустым, значение и адрес его новой вершины.

Очередь

В заданиях Dynamic14–Dynamic28 структура *«очередь»* (queue) моделируется цепочкой связанных узлов-записей типа TNode (см. задание Dynamic2). Поле Next последнего элемента цепочки равно nil. *Началом очереди* («головой», head) считается первый элемент цепочки, *концом* («хвостом», tail) — ее последний элемент. Для возможности быстрого добавления в конец очереди нового элемента удобно хранить, помимо указателя на начало очереди, также и указатель на ее конец. В случае пустой очереди указатели на ее начало и конец полагаются равными nil. Как и для стека, *значением* элемента очереди считается значение его поля Data.

Dynamic14. Дан набор из 10 чисел. Создать очередь, содержащую данные числа в указанном порядке (первое число будет размещаться в начале очереди, последнее — в конце), и вывести указатели P_1 и P_2 на начало и конец очереди.

Dynamic15. Дан набор из 10 чисел. Создать две очереди: первая должна содержать числа из исходного набора с нечетными номерами (1, 3, . . ., 9), а вторая — с четными (2, 4, . . ., 10); порядок чисел в каждой очереди должен совпадать с порядком чисел в исходном наборе. Вывести указатели на начало и конец первой, а затем второй очереди.

Dynamic16. Дан набор из 10 чисел. Создать две очереди: первая должна содержать все нечетные, а вторая — все четные числа из исходного набора (порядок чисел в каждой очереди должен совпадать с порядком чисел в исходном наборе). Вывести указатели на начало и конец первой, а затем второй очереди (одна из очередей может оказаться пустой; в этом случае вывести для нее две константы nil).

Dynamic17. Дано число D и указатели P_1 и P_2 на начало и конец очереди (если очередь является пустой, то $P_1 = P_2 =$ nil). Добавить элемент со значением D в конец очереди и вывести новые адреса начала и конца очереди.

Dynamic18. Дано число D и указатели P_1 и P_2 на начало и конец очереди, содержащей не менее двух элементов. Добавить элемент со значением D в конец очереди и извлечь из очереди первый (начальный) элемент. Вывести значение извлеченного элемента и новые адреса начала и конца очереди. После извлечения элемента из очереди освободить память, занимаемую этим элементом.

Dynamic19. Дано число N (>0) и указатели P_1 и P_2 на начало и конец непустой очереди. Извлечь из очереди N начальных элементов и вывести их значения (если очередь содержит менее N элементов, то извлечь все ее элементы). Вывести также новые адреса начала и конца очереди (для пустой очереди дважды вывести nil). После извлечения элементов из очереди освобождать память, которую они занимали.

Dynamic20. Даны указатели P_1 и P_2 на начало и конец непустой очереди. Извлекать из очереди элементы, пока значение начального элемента очереди не станет четным, и выводить значения извлеченных элементов (если очередь не содержит элементов с четными значениями, то извлечь все ее элементы). Вывести также новые адреса начала и конца очереди (для пустой очереди дважды вывести nil). После извлечения элементов из очереди освобождать память, которую они занимали.

Dynamic21. Даны две очереди; адреса начала и конца первой равны P_1 и P_2, а второй — P_3 и P_4 (если очередь является пустой, то соответствующие

адреса равны nil). Переместить все элементы первой очереди (в порядке от начала к концу) в конец второй очереди и вывести новые адреса начала и конца второй очереди. Операции выделения и освобождения памяти не использовать.

Dynamic22. Дано число N (>0) и две непустые очереди; адреса начала и конца первой равны P_1 и P_2, а второй — P_3 и P_4. Переместить N начальных элементов первой очереди в конец второй очереди. Если первая очередь содержит менее N элементов, то переместить из первой очереди во вторую все элементы. Вывести новые адреса начала и конца первой, а затем второй очереди (для пустой очереди дважды вывести nil). Операции выделения и освобождения памяти не использовать.

Dynamic23. Даны две непустые очереди; адреса начала и конца первой равны P_1 и P_2, а второй — P_3 и P_4. Перемещать элементы из начала первой очереди в конец второй, пока значение начального элемента первой очереди не станет четным (если первая очередь не содержит четных элементов, то переместить из первой очереди во вторую все элементы). Вывести новые адреса начала и конца первой, а затем второй очереди (для пустой очереди дважды вывести nil). Операции выделения и освобождения памяти не использовать.

Dynamic24. Даны две непустые очереди; адреса начала и конца первой равны P_1 и P_2, а второй — P_3 и P_4. Очереди содержат одинаковое количество элементов. Объединить очереди в одну, в которой элементы исходных очередей чередуются (начиная с первого элемента первой очереди). Вывести указатели на начало и конец полученной очереди. Операции выделения и освобождения памяти не использовать.

Dynamic25°. Даны две непустые очереди; адреса начала и конца первой равны P_1 и P_2, а второй — P_3 и P_4. Элементы каждой из очередей упорядочены по возрастанию (в направлении от начала очереди к концу). Объединить очереди в одну с сохранением упорядоченности элементов. Вывести указатели на начало и конец полученной очереди. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic26. Даны указатели P_1 и P_2 на начало и конец очереди (если очередь является пустой, то $P_1 = P_2 =$ nil). Также дано число N (>0) и набор из N чисел. Описать тип TQueue — запись с двумя полями Head и Tail типа PNode (поля указывают на *начало* и *конец* очереди) — и процедуру Enqueue(Q, D), которая добавляет в конец очереди Q новый элемент со

значением D (Q — входной и выходной параметр типа TQueue, D — входной параметр целого типа). С помощью процедуры Enqueue добавить в исходную очередь данный набор чисел и вывести новые адреса ее начала и конца.

Dynamic27. Даны указатели P_1 и P_2 на начало и конец очереди, содержащей не менее пяти элементов. Используя тип TQueue (см. задание Dynamic26), описать функцию Dequeue(Q) целого типа, которая извлекает из очереди первый (начальный) элемент, возвращает его значение и освобождает память, занимаемую извлеченным элементом (Q — входной и выходной параметр типа TQueue). С помощью функции Dequeue извлечь из исходной очереди пять начальных элементов и вывести их значения. Вывести также адреса начала и конца результирующей очереди (если очередь окажется пустой, то эти адреса должны быть равны nil).

Dynamic28. Даны указатели P_1 и P_2 на начало и конец очереди. Используя тип TQueue (см. задание Dynamic26), описать функцию QueueIsEmpty(Q) логического типа, которая возвращает TRUE, если очередь Q пуста, и FALSE в противном случае (Q — входной параметр типа TQueue). Используя эту функцию для проверки состояния очереди, а также функцию Dequeue из задания Dynamic27, извлечь из исходной очереди пять начальных элементов (или все содержащиеся в ней элементы, если их менее пяти) и вывести их значения. Вывести также значение функции QueueIsEmpty для полученной очереди и новые адреса ее начала и конца.

Двусвязный список

Dynamic29. Дан адрес P_2 записи типа TNode, содержащей поле Data (целого типа) и поля Prev и Next (типа PNode — указателя на TNode). Эта запись связана полями Prev и Next соответственно с предыдущей и последующей записью того же типа. Вывести значения полей Data предыдущей и последующей записи, а также адреса P_1 и P_3 предыдущей и последующей записи.

Dynamic30°. Дан указатель P_1 на начало непустой цепочки элементов-записей типа TNode, связанных между собой с помощью поля Next. Используя поле Prev записи TNode, преобразовать исходную (*односвязную*) цепочку в *двусвязную*, в которой каждый элемент связан не только с последующим элементом (с помощью поля Next), но и с предыдущим (с помощью поля Prev). Поле Prev первого элемента положить равным nil. Вывести указа-

тель на последний элемент преобразованной цепочки.

В заданиях Dynamic31–Dynamic69 структура *«двусвязный список»* (doubly linked list) моделируется цепочкой узлов-записей типа TNode, связанных как с предыдущим, так и с последующим узлом (см. задание Dynamic30). Поле Next последнего элемента цепочки и поле Prev первого элемента цепочки равны nil. Для доступа к любому элементу двусвязного списка достаточно иметь указатель на один из его элементов, однако для ускорения операций со списком удобно хранить три указателя: на *первый* элемент списка (first), на его *последний* элемент (last) и на *текущий* элемент (current). Для пустого списка все эти указатели полагаются равными nil. Как в случае стека и очереди, *значением* элемента списка считается значение его поля Data.

Dynamic31. Дан указатель P_0 на один из элементов непустого двусвязного списка. Вывести число N — количество элементов в списке, а также указатели P_1 и P_2 на первый и последний элементы списка.

Dynamic32. Даны числа D_1 и D_2 и указатель P_0 на один из элементов непустого двусвязного списка. Добавить в начало списка новый элемент со значением D_1, а в конец — новый элемент со значением D_2. Вывести адреса первого и последнего элементов полученного списка.

Dynamic33. Дано число D и указатель P_0 на один из элементов непустого двусвязного списка. Вставить перед данным элементом списка новый элемент со значением D и вывести указатель на добавленный элемент списка.

Dynamic34. Дано число D и указатель P_0 на один из элементов непустого двусвязного списка. Вставить после данного элемента списка новый элемент со значением D и вывести указатель на добавленный элемент списка.

Dynamic35. Даны указатели P_1 и P_2 на первый и последний элементы двусвязного списка, содержащего не менее двух элементов. Продублировать в списке первый и последний элементы (новые элементы добавлять перед существующими элементами с такими же значениями) и вывести указатель на первый элемент преобразованного списка.

Dynamic36. Даны указатели P_1 и P_2 на первый и последний элементы двусвязного списка, содержащего не менее двух элементов. Продублировать в списке первый и последний элементы (новые элементы добавлять после существующих элементов с такими же значениями) и вывести указатель на последний элемент преобразованного списка.

Dynamic37. Дан указатель P_1 на первый элемент непустого двусвязного списка. Продублировать в списке все элементы с нечетными номерами (новые элементы добавлять перед существующими элементами с такими же значениями) и вывести указатель на первый элемент преобразованного списка.

Dynamic38. Дан указатель P_1 на первый элемент непустого двусвязного списка. Продублировать в списке все элементы с нечетными номерами (новые элементы добавлять после существующих элементов с такими же значениями) и вывести указатель на последний элемент преобразованного списка.

Dynamic39. Дан указатель P_1 на первый элемент непустого двусвязного списка. Продублировать в списке все элементы с нечетными значениями (новые элементы добавлять перед существующими элементами с такими же значениями) и вывести указатель на первый элемент преобразованного списка.

Dynamic40. Дан указатель P_1 на первый элемент непустого двусвязного списка. Продублировать в списке все элементы с нечетными значениями (новые элементы добавлять после существующих элементов с такими же значениями) и вывести указатель на последний элемент преобразованного списка.

Dynamic41. Дан указатель P_0 на один из элементов непустого двусвязного списка. Удалить из списка данный элемент и вывести два указателя: на элемент, предшествующий удаленному, и на элемент, следующий за удаленным (один или оба этих элемента могут отсутствовать; для отсутствующих элементов выводить nil). После удаления элемента из списка освободить память, занимаемую этим элементом.

Dynamic42. Дан указатель P_1 на первый элемент двусвязного списка, содержащего не менее двух элементов. Удалить из списка все элементы с нечетными номерами и вывести указатель на первый элемент преобразованного списка. После удаления элементов из списка освобождать память, которую они занимали.

Dynamic43. Дан указатель P_1 на первый элемент непустого двусвязного списка. Удалить из списка все элементы с нечетными значениями и вывести указатель на первый элемент преобразованного списка (если в результате удаления элементов список окажется пустым, то вывести nil). После удаления элементов из списка освобождать память, которую они занимали.

Dynamic44. Дан указатель P_0 на один из элементов непустого двусвязного списка. Переместить данный элемент в конец списка и вывести указатели на первый и последний элементы преобразованного списка. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic45. Дан указатель P_0 на один из элементов непустого двусвязного списка. Переместить данный элемент в начало списка и вывести указатели на первый и последний элементы преобразованного списка. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic46. Дано число K (>0) и указатель P_0 на один из элементов непустого двусвязного списка. Переместить в списке данный элемент на K позиций вперед (если после данного элемента находится менее K элементов, то переместить его в конец списка). Вывести указатели на первый и последний элементы преобразованного списка. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic47. Дано число K (>0) и указатель P_0 на один из элементов непустого двусвязного списка. Переместить в списке данный элемент на K позиций назад (если перед данным элементом находится менее K элементов, то переместить его в начало списка). Вывести указатели на первый и последний элементы преобразованного списка. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic48. Даны указатели P_X и P_Y на два различных элемента двусвязного списка (элемент с адресом P_X находится в списке перед элементом с адресом P_Y, но не обязательно рядом с ним). Поменять местами данные элементы и вывести указатель на первый элемент преобразованного списка. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic49°. Дан указатель P_1 на первый элемент непустого двусвязного списка. Перегруппировать его элементы, переместив все элементы с нечетными номерами в конец списка (в том же порядке) и вывести указатель на первый элемент преобразованного списка. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic50. Дан указатель P_1 на первый элемент непустого двусвязного списка. Перегруппировать его элементы, переместив все элементы с нечетными значениями в конец списка (в том же порядке) и вывести указатель на

первый элемент преобразованного списка. Операции выделения и освобождения памяти не использовать, поля Data не изменять.

Dynamic51. Даны два непустых двусвязных списка и связанные с ними указатели: P_1 и P_2 указывают на первый и последний элементы первого списка, P_0 — на один из элементов второго. Объединить исходные списки, поместив все элементы первого списка (в том же порядке) перед данным элементом второго списка, и вывести указатели на первый и последний элементы объединенного списка. Операции выделения и освобождения памяти не использовать.

Dynamic52. Даны два непустых двусвязных списка и связанные с ними указатели: P_1 и P_2 указывают на первый и последний элементы первого списка, P_0 — на один из элементов второго. Объединить исходные списки, поместив все элементы первого списка (в том же порядке) после данного элемента второго списка, и вывести указатели на первый и последний элементы объединенного списка. Операции выделения и освобождения памяти не использовать.

Dynamic53. Даны указатели P_X и P_Y на два различных элемента двусвязного списка; элемент с адресом P_X находится в списке перед элементом с адресом P_Y, но не обязательно рядом с ним. Переместить элементы, расположенные между данными элементами (включая данные элементы), в новый список (в том же порядке). Вывести указатели на первые элементы преобразованного и нового списков. Если преобразованный список окажется пустым, то связанный с ним указатель положить равным nil. Операции выделения и освобождения памяти не использовать.

Dynamic54. Даны указатели P_X и P_Y на два различных элемента двусвязного списка; элемент с адресом P_X находится в списке перед элементом с адресом P_Y, но не обязательно рядом с ним. Переместить элементы, расположенные между данными элементами (не включая данные элементы), в новый список (в том же порядке). Вывести указатели на первые элементы преобразованного и нового списков. Если новый список окажется пустым, то связанный с ним указатель положить равным nil. Операции выделения и освобождения памяти не использовать.

Dynamic55°. Дан указатель P_1 на первый элемент непустого двусвязного списка. Преобразовать список в *циклический*, записав в поле Next последнего элемента списка адрес его первого элемента, а в поле Prev первого элемента — адрес последнего элемента. Вывести указатель на элемент, который

был последним элементом исходного списка.

Dynamic56. Даны указатели P_1 и P_2 на первый и последний элементы непустого двусвязного списка, содержащего четное количество элементов. Преобразовать список в два *циклических* списка (см. задание Dynamic55), первый из которых содержит первую половину элементов исходного списка, а второй — вторую половину. Вывести указатели P_3 и P_4 на два средних элемента исходного списка (элемент с адресом P_3 должен входить в первый циклический список, а элемент с адресом P_4 — во второй). Операции выделения и освобождения памяти не использовать.

Dynamic57. Дано число K (>0) и указатели P_1 и P_2 на первый и последний элементы непустого двусвязного списка. Осуществить *циклический сдвиг* элементов списка на K позиций вперед (то есть в направлении от начала к концу списка) и вывести указатели на первый и последний элементы полученного списка. Для выполнения циклического сдвига преобразовать исходный список в циклический (см. задание Dynamic55), после чего «разорвать» его в позиции, соответствующей данному значению K. Операции выделения и освобождения памяти не использовать.

Dynamic58. Дано число K (>0) и указатели P_1 и P_2 на первый и последний элементы непустого двусвязного списка. Осуществить *циклический сдвиг* элементов списка на K позиций назад (то есть в направлении от конца к началу списка) и вывести указатели на первый и последний элементы полученного списка. Для выполнения циклического сдвига преобразовать исходный список в циклический (см. задание Dynamic55), после чего «разорвать» его в позиции, соответствующей данному значению K. Операции выделения и освобождения памяти не использовать.

Dynamic59°. Даны указатели P_1, P_2 и P_3 на первый, последний и текущий элементы двусвязного списка (если список является пустым, то $P_1 = P_2 = P_3 = \text{nil}$). Также дано число N (>0) и набор из N чисел. Описать тип TList — запись с полями First, Last и Current типа PNode (поля указывают соответственно на *первый*, *последний* и *текущий* элементы списка) — и процедуру InsertLast(L, D), которая добавляет новый элемент со значением D в конец списка L (L — входной и выходной параметр типа TList, D — входной параметр целого типа). Добавленный элемент становится текущим. С помощью этой процедуры добавить в конец исходного списка данный набор чисел (в том же порядке) и вывести новые адреса его первого, последнего и текущего элементов.

Dynamic60. Даны указатели P_1, P_2 и P_3 на первый, последний и текущий элементы двусвязного списка (если список является пустым, то $P_1 = P_2 = P_3 = \text{nil}$). Также дано число N (> 0) и набор из N чисел. Используя тип TList (см. задание Dynamic59), описать процедуру InsertFirst(L, D), которая добавляет новый элемент со значением D в начало списка L (L — входной и выходной параметр типа TList, D — входной параметр целого типа). Добавленный элемент становится текущим. С помощью этой процедуры добавить в начало исходного списка данный набор чисел (добавленные числа будут располагаться в списке в обратном порядке) и вывести новые адреса его первого, последнего и текущего элементов.

Dynamic61. Дан непустой двусвязный список, первый, последний и текущий элементы которого имеют адреса P_1, P_2 и P_3. Также даны пять чисел. Используя тип TList (см. задание Dynamic59), описать процедуру InsertBefore(L, D), которая вставляет новый элемент со значением D перед текущим элементом списка L (L — входной и выходной параметр типа TList, D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью этой процедуры вставить пять данных чисел в исходный список и вывести новые адреса его первого, последнего и текущего элементов.

Dynamic62. Дан непустой двусвязный список, первый, последний и текущий элементы которого имеют адреса P_1, P_2 и P_3. Также даны пять чисел. Используя тип TList (см. задание Dynamic59), описать процедуру InsertAfter(L, D), которая вставляет новый элемент со значением D после текущего элемента списка L (L — входной и выходной параметр типа TList, D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью этой процедуры вставить пять данных чисел в исходный список и вывести новые адреса его первого, последнего и текущего элементов.

Dynamic63°. Дан непустой двусвязный список, первый, последний и текущий элементы которого имеют адреса P_1, P_2 и P_3. Используя тип TList (см. задание Dynamic59), описать процедуры ToFirst(L) (делает текущим первый элемент списка L), ToNext(L) (делает текущим в списке L следующий элемент, если он существует), SetData(L, D) (присваивает текущему элементу списка L значение D целого типа) и функцию IsLast(L) логического типа (возвращает TRUE, если текущий элемент списка L является его последним элементом, и FALSE в противном случае). Параметр L имеет тип TList;

в процедурах ToFirst и ToNext он является входным и выходным. С помощью этих процедур и функций присвоить нулевые значения элементам исходного списка с нечетными номерами и вывести количество элементов в списке, а также новые адреса его первого, последнего и текущего элементов.

Dynamic64. Дан непустой двусвязный список, первый, последний и текущий элементы которого имеют адреса P_1, P_2 и P_3. Используя тип TList (см. задание Dynamic59), описать процедуры ToLast(L) (делает текущим последний элемент списка L), ToPrev(L) (делает текущим в списке L предыдущий элемент, если он существует) и функции GetData(L) целого типа (возвращает значение текущего элемента списка L), IsFirst(L) логического типа (возвращает TRUE, если текущий элемент списка L является его первым элементом, и FALSE в противном случае). Параметр L имеет тип TList; в процедурах ToLast и ToPrev он является входным и выходным. С помощью этих процедур и функций вывести все четные значения элементов исходного списка, просматривая список с конца. Вывести также количество элементов в списке.

Dynamic65. Даны указатели P_1, P_2 и P_3 на первый, последний и текущий элементы двусвязного списка, содержащего не менее пяти элементов. Используя тип TList (см. задание Dynamic59), описать функцию DeleteCurrent(L) целого типа, удаляющую из списка L текущий элемент и возвращающую его значение (L — входной и выходной параметр типа TList). После удаления элемента текущим становится следующий элемент или, если следующего элемента не существует, последний элемент списка. Функция также освобождает память, занимаемую удаленным элементом. С помощью этой функции удалить из исходного списка пять элементов и вывести их значения. Вывести также новые адреса первого, последнего и текущего элементов списка (для пустого списка вывести три константы nil).

Dynamic66. Даны указатели P_1, P_2 и P_3 на первый, последний и текущий элементы непустого двусвязного списка. Используя тип TList (см. задание Dynamic59), описать процедуру SplitList(L_1, L_2), которая переносит элементы списка L_1 от текущего до последнего в новый список L_2 (таким образом, список L_1 делится на две части, причем первая часть может оказаться пустой). Параметры процедуры имеют тип TList; первый параметр является входным и выходным, второй — выходным. Текущими элемен-

тами непустых результирующих списков становятся их первые элементы. Операции выделения и освобождения памяти в процедуре не использовать. С помощью этой процедуры разбить исходный список на два и вывести адреса первого, последнего и текущего элементов полученных списков.

Dynamic67. Даны указатели на первый, последний и текущий элементы двух непустых двусвязных списков. Используя тип TList (см. задание Dynamic59), описать процедуру AddList(L_1, L_2), которая добавляет все элементы из списка L_1 (в том же порядке) в конец списка L_2; в результате список L_1 становится пустым. Текущим элементом списка L_2 становится первый из добавленных элементов. Оба параметра процедуры имеют тип TList и являются входными и выходными. Операции выделения и освобождения памяти в процедуре не использовать. С помощью этой процедуры добавить первый из исходных списков в конец второго и вывести адреса первого, последнего и текущего элементов объединенного списка.

Dynamic68. Даны указатели на первый, последний и текущий элементы двух непустых двусвязных списков. Используя тип TList (см. задание Dynamic59), описать процедуру InsertList(L_1, L_2), которая вставляет все элементы из списка L_1 (в том же порядке) в список L_2 перед его текущим элементом; в результате список L_1 становится пустым. Текущим элементом списка L_2 становится первый из вставленных элементов. Оба параметра процедуры имеют тип TList и являются входными и выходными. Операции выделения и освобождения памяти в процедуре не использовать. С помощью этой процедуры вставить первый из исходных списков в текущую позицию второго и вывести адреса первого, последнего и текущего элементов объединенного списка.

Dynamic69. Даны указатели на первый, последний и текущий элементы двух двусвязных списков (второй список может быть пустым). Используя тип TList (см. задание Dynamic59), описать процедуру MoveCurrent(L_1, L_2), которая перемещает текущий элемент списка L_1 в список L_2 (элемент вставляется после текущего элемента списка L_2 и сам становится текущим; в списке L_1 текущим становится следующий элемент или, если следующего элемента не существует, последний элемент). Оба параметра процедуры имеют тип TList и являются входными и выходными. Операции выделения и освобождения памяти в процедуре не использовать. С помощью этой процедуры переместить текущий элемент первого списка

во второй и вывести адреса первого, последнего и текущего элементов полученных списков.

Список с барьерным элементом

Использованная в заданиях Dynamic31–Dynamic69 реализация двусвязного списка в виде цепочки узлов, ограниченной по краям нулевыми указателями, не является единственно возможной. Двусвязный список можно также реализовать в виде *замкнутой* цепочки узлов с дополнительным *фиктивным*, или *барьерным*, элементом. Этот барьерный элемент связан своими полями Next и Prev с первым и последним «настоящим» элементом списка соответственно, поэтому, имея указатель на барьерный элемент, можно сразу перейти как к первому, так и к последнему элементу списка (естественно, первый и последний элементы также связаны с барьерным элементом своими полями Prev и Next соответственно). Для работы с двусвязным списком, снабженным барьерным элементом, достаточно хранить два указателя: Barrier, указывающий на барьерный элемент, и Current, указывающий на текущий элемент (который может быть как «настоящим», так и барьерным элементом). Поле Data барьерного элемента может быть произвольным; для определенности его можно положить равным 0. *Пустой список* в данной реализации представляет собой единственный барьерный элемент, «замкнутый на себя».

Задания Dynamic70–Dynamic80 посвящены *двусвязным спискам с барьерным элементом*.

Dynamic70°. Даны указатели P_1 и P_2 на первый и последний элементы двусвязного списка, реализованного в виде цепочки узлов, ограниченной по краям нулевыми указателями (если список пуст, то $P_1 = P_2 = \text{nil}$). Преобразовать исходный список в *циклический список* (см. задание Dynamic55), снабженный *барьерным элементом*. Барьерный элемент должен иметь значение 0 и быть связан своими полями Next и Prev с первым и последним элементом исходного списка (в случае пустого исходного списка поля Next и Prev барьерного элемента должны указывать на сам барьерный элемент). Вывести указатель на барьерный элемент полученного списка. Операцию выделения памяти использовать только для создания барьерного элемента.

Dynamic71. Даны указатели P_1 и P_2 на барьерный и текущий элементы двусвязного списка (о списке с барьерным элементом см. задание Dynamic70).

Разбить список на два, перенеся во второй список все элементы от текущего до последнего и добавив ко второму списку барьерный элемент. Если текущий элемент исходного списка является барьерным элементом, то второй список должен быть пустым (то есть состоять только из барьерного элемента). Вывести указатель на барьерный элемент второго списка. Операцию выделения памяти использовать только для создания барьерного элемента второго списка.

Dynamic72. Даны указатели P_1 и P_2 на барьерные элементы двух двусвязных списков (о списке с барьерным элементом см. задание Dynamic70). Объединить исходные списки, связав конец первого и начало второго списка (барьерным элементом объединенного списка должен остаться барьерный элемент первого списка). Вывести указатели на первый и последний элементы объединенного списка (если объединенный список является пустым, то дважды вывести указатель на его барьерный элемент). После удаления лишнего барьерного элемента освободить занимаемую им память.

Dynamic73. Даны указатели P_1 и P_2 на барьерные элементы двух двусвязных списков (о списке с барьерным элементом см. задание Dynamic70). Объединить исходные списки, связав конец первого и начало второго списка (барьерным элементом объединенного списка должен остаться барьерный элемент второго списка). Вывести указатели на первый и последний элементы объединенного списка (если объединенный список является пустым, то дважды вывести указатель на его барьерный элемент). После удаления лишнего барьерного элемента освободить занимаемую им память.

Dynamic74°. Даны указатели P_1 и P_2 на барьерный и текущий элементы двусвязного списка (о списке с барьерным элементом см. задание Dynamic70). Также дано число N (>0) и набор из N чисел. Описать тип TListB — запись с полями Barrier и Current типа PNode (поля указывают соответственно на *барьерный* и *текущий* элементы списка) — и процедуру LBInsertLast(L, D), которая добавляет новый элемент со значением D в конец списка L (L — входной и выходной параметр типа TListB, D — входной параметр целого типа). Добавленный элемент становится текущим. С помощью этой процедуры добавить в конец исходного списка данный набор чисел (в том же порядке) и вывести адрес текущего элемента полученного списка.

Dynamic75. Даны указатели P_1 и P_2 на барьерный и текущий элемен-

ты двусвязного списка. Также дано число N (>0) и набор из N чисел. Используя тип TListB (см. задание Dynamic74), описать процедуру LBInsertFirst(L, D), которая добавляет новый элемент со значением D в начало списка L (L — входной и выходной параметр типа TListB, D — входной параметр целого типа). Добавленный элемент становится текущим. С помощью этой процедуры добавить в начало исходного списка данный набор чисел (добавленные числа будут располагаться в списке в обратном порядке) и вывести адрес текущего элемента полученного списка.

Dynamic76. Даны указатели P_1 и P_2 на барьерный и текущий элементы двусвязного списка. Также даны пять чисел. Используя тип TListB (см. задание Dynamic74), описать процедуру LBInsertBefore(L, D), которая вставляет новый элемент со значением D перед текущим элементом списка L (L — входной и выходной параметр типа TListB, D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью этой процедуры вставить пять данных чисел в исходный список и вывести новый адрес его текущего элемента.

Dynamic77. Даны указатели P_1 и P_2 на барьерный и текущий элементы двусвязного списка. Также даны пять чисел. Используя тип TListB (см. задание Dynamic74), описать процедуру LBInsertAfter(L, D), которая вставляет новый элемент со значением D после текущего элемента списка L (L — входной и выходной параметр типа TListB, D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью этой процедуры вставить пять данных чисел в исходный список и вывести новый адрес его текущего элемента.

Dynamic78°. Даны указатели P_1 и P_2 на барьерный и текущий элементы двусвязного списка. Используя тип TListB (см. задание Dynamic74), описать процедуры LBToFirst(L) (делает текущим первый элемент списка L), LBToNext(L) (делает текущим в списке L следующий элемент), LBSetData(L, D) (присваивает текущему элементу списка L значение D целого типа, если данный элемент не является барьерным) и функцию IsBarrier(L) логического типа (возвращает TRUE, если текущий элемент списка L является его барьерным элементом, и FALSE в противном случае). Параметр L имеет тип TListB; в процедурах LBToFirst и LBToNext он является входным и выходным. С помощью этих процедур и функций присвоить нулевые значения элементам исходного списка с нечетными номерами и вывести количество элементов в списке, а также новый ад-

рес текущего элемента списка. Нумерация ведется от первого элемента списка; барьерный элемент не нумеруется и не учитывается при подсчете элементов.

Dynamic79. Даны указатели P_1 и P_2 на барьерный и текущий элементы двусвязного списка. Используя тип TListB (см. задание Dynamic74), описать процедуры LBToLast(L) (делает текущим последний элемент списка L), LBToPrev(L) (делает текущим в списке L предыдущий элемент) и функцию LBGetData(L) целого типа (возвращает значение текущего элемента списка L). Параметр L имеет тип TListB; в процедурах LBToLast и LBToPrev он является входным и выходным. С помощью этих процедур и функций, а также с использованием функции IsBarrier из задания Dynamic78, вывести все четные значения элементов исходного списка, просматривая список с конца. Вывести также количество элементов в списке. Барьерный элемент не обрабатывается и не учитывается при подсчете элементов.

Dynamic80. Даны указатели P_1 и P_2 на барьерный и текущий элементы непустого двусвязного списка, причем текущий элемент не совпадает с барьерным. Используя тип TListB (см. задание Dynamic74), описать функцию LBDeleteCurrent(L) целого типа, удаляющую из списка L текущий элемент и возвращающую его значение (L — входной и выходной параметр типа TListB). Текущим становится следующий элемент или, если следующий элемент является барьерным, предыдущий элемент списка. Функция также освобождает память, занимаемую удаленным элементом. Если текущим элементом является барьерный элемент, то функция не выполняет никаких действий и возвращает 0. С помощью этой функции, а также функции IsBarrier из задания Dynamic78, удалить из исходного списка пять элементов (или все элементы, если их менее пяти) и вывести их значения. Вывести также новый адрес текущего элемента списка.

Динамические структуры данных (.NET)

Данный раздел содержит описание варианта группы Dynamic, предназначенного для языков программирования платформы .NET: C# и Visual Basic .NET.

Все числа, упоминаемые в заданиях данной группы, являются целыми. Все объекты имеют тип Node. Данный классовый тип определен в задачнике

Programming Taskbook (точнее, в сборке pt4net.dll, определяющей пространство имен PT4) и включает следующие открытые свойства и методы:

конструкторы:
```
public Node();
public Node(int aData);
public Node(int aData, Node aNext);
public Node(int aData, Node aNext, Node aPrev);
```
свойства (доступны для чтения и для записи):
```
public int Data;
public Node Next;
public Node Prev;
```
метод, освобождающий неуправляемые ресурсы, используемые объектом:
```
public void Dispose();
```

Во вводных заданиях Dynamic1–Dynamic2, а также в заданиях на стек и очередь (Dynamic3–Dynamic28) при работе с объектами типа Node свойство Prev не используется (см. задание Dynamic1); в заданиях на списки (Dynamic29–Dynamic80) используются все свойства объектов типа Node (см. задание Dynamic29).

Так как в языках платформы .NET принята ссылочная объектная модель, то есть любая переменная объектного типа является *ссылкой* на ту область памяти, в которой фактически размещен данный объект, выражение «вывести ссылку на объект» в формулировках заданий всегда означает, что требуется вывести значение переменной типа Node, связанной с этим объектом (используя метод Put класса PT, описанного в сборке pt4net.dll).

В заданиях, в которых идет речь о номерах элементов списка, предполагается, что элементы списка нумеруются от 1.

Для обозначения пустого объекта в формулировках заданий используется имя null (как в языке C#).

Dynamic1. Дан объект A_1 типа Node, имеющий открытые свойства Data целого типа и Next типа Node. Свойство Next данного объекта содержит ссылку на объект A_2 (того же типа Node). Вывести значения свойств Data обоих объектов, а также ссылку на объект A_2.

Dynamic2°. Дан объект A_1 типа Node. Этот объект связан своим свойством Next со следующим объектом типа Node, он, в свою очередь, — со сле-

дующим, и так далее до объекта со свойством Next, равным null (таким образом, возникает *цепочка* связанных объектов). Вывести значения свойств Data для всех элементов цепочки, *длину цепочки* (то есть число ее элементов) и ссылку на ее последний элемент.

Стек

В заданиях Dynamic3–Dynamic13 структура *«стек»* (stack) моделируется цепочкой связанных узлов-объектов типа Node (см. задание Dynamic2). Свойство Next последнего элемента цепочки равно null. *Вершиной стека* (top) считается первый элемент цепочки. Для доступа к стеку используется переменная типа Node — ссылка на вершину стека (для пустого стека данная переменная полагается равной null). *Значением* элемента стека считается значение его свойства Data.

Dynamic3°. Дано число D и вершина A_1 непустого стека. Добавить элемент со значением D в стек и вывести ссылку A_2 на новую вершину стека.

Dynamic4. Дано число N (>0) и набор из N чисел. Создать стек, содержащий исходные числа (последнее число будет вершиной стека), и вывести ссылку на его вершину.

Dynamic5°. Дана вершина A_1 непустого стека. Извлечь из стека первый (верхний) элемент и вывести его значение D, а также ссылку A_2 на новую вершину стека. Если после извлечения элемента стек окажется пустым, то положить A_2 = null. После извлечения элемента из стека освободить ресурсы, используемые этим элементом, вызвав его метод Dispose.

Dynamic6. Дана вершина A_1 стека, содержащего не менее десяти элементов. Извлечь из стека первые девять элементов и вывести их значения. Вывести также ссылку на новую вершину стека. После извлечения элементов из стека освобождать ресурсы, которые они использовали, вызывая для этих элементов метод Dispose.

Dynamic7. Дана вершина A_1 стека (если стек пуст, то A_1 = null). Извлечь из стека все элементы и вывести их значения. Вывести также количество извлеченных элементов N (для пустого стека вывести 0). После извлечения элементов из стека освобождать ресурсы, которые они использовали, вызывая для этих элементов метод Dispose.

Dynamic8°. Даны вершины A_1 и A_2 двух непустых стеков. Переместить все элементы из первого стека во второй (в результате элементы первого стека

будут располагаться во втором стеке в порядке, обратном исходному) и вывести ссылку на новую вершину второго стека. Новые объекты типа Node не создавать.

Dynamic9°. Даны вершины A_1 и A_2 двух непустых стеков. Перемещать элементы из первого стека во второй, пока значение вершины первого стека не станет четным (перемещенные элементы первого стека будут располагаться во втором стеке в порядке, обратном исходному). Если в первом стеке нет элементов с четными значениями, то переместить из первого стека во второй все элементы. Вывести ссылки на новые вершины первого и второго стека (если первый стек окажется пустым, то вывести для него константу null). Новые объекты типа Node не создавать.

Dynamic10°. Дана вершина A_1 непустого стека. Создать два новых стека, переместив в первый из них все элементы исходного стека с четными значениями, а во второй — с нечетными (элементы в новых стеках будут располагаться в порядке, обратном исходному; один из этих стеков может оказаться пустым). Вывести ссылки на вершины полученных стеков (для пустого стека вывести константу null). Новые объекты типа Node не создавать.

Dynamic11°. Дана вершина A_1 стека (если стек пуст, то A_1 = null). Также дано число N (>0) и набор из N чисел. Описать класс IntStack, содержащий следующие члены:

- закрытое поле top типа Node (*вершина* стека);
- конструктор с параметром aTop — вершиной существующего стека;
- процедура Push(D), которая добавляет в стек новый элемент со значением D (D — входной параметр целого типа);
- процедура Put (без параметров), которая выводит ссылку на поле top, используя метод Put класса PT.

С помощью метода Push добавить в исходный стек данный набор чисел (последнее число будет вершиной стека) и вывести ссылку на новую вершину стека, используя для этого метод Put класса IntStack.

Dynamic12°. Дана вершина A_1 стека, содержащего не менее пяти элементов. Включить в класс IntStack (см. задание Dynamic11) функцию Pop целого типа (без параметров), которая извлекает из стека первый (верхний) элемент, возвращает его значение и вызывает для него метод Dispose. С помощью метода Pop извлечь из исходного стека пять элементов и вывести их значения. Вывести также ссылку на новую вершину стека (если

результирующий стек окажется пустым, то эта ссылка должна быть равна null).

Dynamic13. Дана вершина A_1 стека. Включить в класс IntStack (см. задание Dynamic11) две функции: IsEmpty логического типа (возвращает TRUE, если стек пуст, и FALSE в противном случае) и Peek целого типа (возвращает значение вершины непустого стека, не удаляя ее из стека). Функции не имеют параметров. С помощью этих функций, а также функции Pop из задания Dynamic12, извлечь из исходного стека пять элементов (или все содержащиеся в нем элементы, если их менее пяти) и вывести их значения. Вывести также значение функции IsEmpty для результирующего стека и, если результирующий стек не является пустым, значение его новой вершины и ссылку на нее.

Очередь

В заданиях Dynamic14–Dynamic28 структура *«очередь»* (queue) моделируется цепочкой связанных узлов-объектов типа Node (см. задание Dynamic2). Свойство Next последнего элемента цепочки равно null. *Началом очереди* («головой», head) считается первый элемент цепочки, *концом* («хвостом», tail) — ее последний элемент. Для возможности быстрого добавления в конец очереди нового элемента удобно хранить, помимо ссылки на начало очереди, также и ссылку на ее конец. В случае пустой очереди ссылки на ее начало и конец полагаются равными null. Как и для стека, *значением* элемента очереди считается значение его свойства Data.

Dynamic14. Дан набор из 10 чисел. Создать очередь, содержащую данные числа в указанном порядке (первое число будет размещаться в начале очереди, последнее — в конце), и вывести ссылки A_1 и A_2 на начало и конец очереди.

Dynamic15. Дан набор из 10 чисел. Создать две очереди: первая должна содержать числа из исходного набора с нечетными номерами (1, 3, …, 9), а вторая — с четными (2, 4, …, 10); порядок чисел в каждой очереди должен совпадать с порядком чисел в исходном наборе. Вывести ссылки на начало и конец первой, а затем второй очереди.

Dynamic16. Дан набор из 10 чисел. Создать две очереди: первая должна содержать все нечетные, а вторая — все четные числа из исходного набора (порядок чисел в каждой очереди должен совпадать с порядком чисел в

исходном наборе). Вывести ссылки на начало и конец первой, а затем второй очереди (одна из очередей может оказаться пустой; в этом случае вывести для нее две константы null).

Dynamic17. Дано число D и ссылки A_1 и A_2 на начало и конец очереди (если очередь является пустой, то $A_1 = A_2 =$ null). Добавить элемент со значением D в конец очереди и вывести ссылки на начало и конец полученной очереди.

Dynamic18. Дано число D и ссылки A_1 и A_2 на начало и конец очереди, содержащей не менее двух элементов. Добавить элемент со значением D в конец очереди и извлечь из очереди первый (начальный) элемент. Вывести значение извлеченного элемента, а также ссылки на начало и конец полученной очереди. После извлечения элемента из очереди вызвать для него метод Dispose.

Dynamic19. Дано число N (>0) и ссылки A_1 и A_2 на начало и конец непустой очереди. Извлечь из очереди N начальных элементов и вывести их значения (если очередь содержит менее N элементов, то извлечь все ее элементы). Вывести также ссылки на начало и конец полученной очереди (для пустой очереди дважды вывести null). После извлечения элементов вызывать для них метод Dispose.

Dynamic20. Даны ссылки A_1 и A_2 на начало и конец непустой очереди. Извлекать из очереди элементы, пока значение начального элемента очереди не станет четным, и выводить значения извлеченных элементов (если очередь не содержит элементов с четными значениями, то извлечь все ее элементы). Вывести также ссылки на начало и конец полученной очереди (для пустой очереди дважды вывести null). После извлечения элементов вызывать для них метод Dispose.

Dynamic21. Даны две очереди; начало и конец первой равны A_1 и A_2, а второй — A_3 и A_4 (если очередь является пустой, то соответствующие объекты равны null). Переместить все элементы первой очереди (в порядке от начала к концу) в конец второй очереди и вывести ссылки на начало и конец преобразованной второй очереди. Новые объекты типа Node не создавать.

Dynamic22. Дано число N (>0) и две непустые очереди; начало и конец первой равны A_1 и A_2, а второй — A_3 и A_4. Переместить N начальных элементов первой очереди в конец второй очереди. Если первая очередь содержит менее N элементов, то переместить из первой очереди во вторую все эле-

менты. Вывести новые ссылки на начало и конец первой, а затем второй очереди (для пустой очереди дважды вывести null). Новые объекты типа Node не создавать.

Dynamic23. Даны две непустые очереди; начало и конец первой равны A_1 и A_2, а второй — A_3 и A_4. Перемещать элементы из начала первой очереди в конец второй, пока значение начального элемента первой очереди не станет четным (если первая очередь не содержит четных элементов, то переместить из первой очереди во вторую все элементы). Вывести новые ссылки на начало и конец первой, а затем второй очереди (для пустой очереди дважды вывести null). Новые объекты типа Node не создавать.

Dynamic24. Даны две непустые очереди; начало и конец первой равны A_1 и A_2, а второй — A_3 и A_4. Очереди содержат одинаковое количество элементов. Объединить очереди в одну, в которой элементы исходных очередей чередуются (начиная с первого элемента первой очереди). Вывести ссылки на начало и конец полученной очереди. Новые объекты типа Node не создавать.

Dynamic25°. Даны две непустые очереди; начало и конец первой равны A_1 и A_2, а второй — A_3 и A_4. Элементы каждой из очередей упорядочены по возрастанию (в направлении от начала очереди к концу). Объединить очереди в одну с сохранением упорядоченности элементов. Вывести ссылки на начало и конец полученной очереди. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic26. Даны ссылки A_1 и A_2 на начало и конец очереди (если очередь является пустой, то $A_1 = A_2$ = null). Также дано число N (>0) и набор из N чисел. Описать класс IntQueue, содержащий следующие члены:

- закрытые поля head и tail типа Node (*начало* и *конец* очереди);
- конструктор с параметрами aHead, aTail — началом и концом существующей очереди;
- процедура Enqueue(D), которая добавляет в конец очереди новый элемент со значением D (D — входной параметр целого типа);
- процедура Put (без параметров), которая выводит ссылки на поля head и tail, используя метод Put класса PT.

С помощью метода Enqueue добавить в исходную очередь данный набор чисел и вывести новые ссылки на ее начало и конец, используя для этого метод Put класса IntQueue.

Dynamic27. Даны ссылки A_1 и A_2 на начало и конец очереди, содержа-

щей не менее пяти элементов. Включить в класс IntQueue (см. задание Dynamic26) функцию Dequeue целого типа (без параметров), которая извлекает из очереди первый (начальный) элемент, возвращает его значение и вызывает для него метод Dispose. С помощью функции Dequeue извлечь из исходной очереди пять начальных элементов и вывести их значения. Вывести также ссылки на начало и конец результирующей очереди (если очередь окажется пустой, то эти ссылки должны быть равны null).

Dynamic28. Даны ссылки A_1 и A_2 на начало и конец очереди. Включить в класс IntQueue (см. задание Dynamic26) функцию IsEmpty логического типа (без параметров), которая возвращает TRUE, если очередь пуста, и FALSE в противном случае. Используя эту функцию для проверки состояния очереди, а также функцию Dequeue из задания Dynamic27, извлечь из исходной очереди пять начальных элементов (или все содержащиеся в ней элементы, если их менее пяти) и вывести их значения. Вывести также значение функции IsEmpty для полученной очереди и новые ссылки на ее начало и конец.

Двусвязный список

Dynamic29. Дан объект A_2 типа Node, имеющий открытое свойство Data целого типа, а также открытые свойства Prev и Next типа Node. Свойство Prev объекта A_2 содержит ссылку на предыдущий объект того же типа, а свойство Next — ссылку на следующий объект. Вывести значения свойств Data предыдущего и следующего объекта, а также ссылки A_1 и A_3 на предыдущий и следующий объект.

Dynamic30°. Дана ссылка A_1 на начало непустой цепочки элементов-объектов типа Node, связанных между собой с помощью своих свойств Next. Используя свойства Prev данных объектов, преобразовать исходную (*односвязную*) цепочку в *двусвязную*, в которой каждый элемент связан не только с последующим элементом (с помощью свойства Next), но и с предыдущим (с помощью свойства Prev). Свойство Prev первого элемента положить равным null. Вывести ссылку A_2 на последний элемент преобразованной цепочки.

В заданиях Dynamic31–Dynamic69 структура *«двусвязный список»* (doubly linked list) моделируется цепочкой узлов-объектов типа Node, связанных как с предыдущим, так и с последующим узлом (см. задание Dynamic30). Свойство

Next последнего элемента цепочки и свойство Prev первого элемента цепочки равны null. Для доступа к любому элементу двусвязного списка достаточно иметь ссылку на один из его элементов, однако для ускорения операций со списком удобно хранить три ссылки: на *первый* элемент списка (first), на его *последний* элемент (last) и на *текущий* элемент (current). Для пустого списка все эти ссылки полагаются равными null. Как в случае стека и очереди, *значением* элемента списка считается значение его свойства Data.

Dynamic31. Дана ссылка A_0 на один из элементов непустого двусвязного списка. Вывести число N — количество элементов в списке, а также ссылки A_1 и A_2 на первый и последний элементы списка.

Dynamic32. Даны числа D_1 и D_2 и ссылка A_0 на один из элементов непустого двусвязного списка. Добавить в начало списка новый элемент со значением D_1, а в конец — новый элемент со значением D_2. Вывести ссылки на первый и последний элементы полученного списка.

Dynamic33. Дано число D и ссылка A_0 на один из элементов непустого двусвязного списка. Вставить перед данным элементом списка новый элемент со значением D и вывести ссылку на добавленный элемент списка.

Dynamic34. Дано число D и ссылка A_0 на один из элементов непустого двусвязного списка. Вставить после данного элемента списка новый элемент со значением D и вывести ссылку на добавленный элемент списка.

Dynamic35. Даны ссылки A_1 и A_2 на первый и последний элементы двусвязного списка, содержащего не менее двух элементов. Продублировать в списке первый и последний элементы (новые элементы добавлять перед существующими элементами с такими же значениями) и вывести ссылку на первый элемент преобразованного списка.

Dynamic36. Даны ссылки A_1 и A_2 на первый и последний элементы двусвязного списка, содержащего не менее двух элементов. Продублировать в списке первый и последний элементы (новые элементы добавлять после существующих элементов с такими же значениями) и вывести ссылку на последний элемент преобразованного списка.

Dynamic37. Дан первый элемент A_1 непустого двусвязного списка. Продублировать в списке все элементы с нечетными номерами (новые элементы добавлять перед существующими элементами с такими же значениями) и вывести ссылку на первый элемент преобразованного списка.

Dynamic38. Дан первый элемент A_1 непустого двусвязного списка. Продублировать в списке все элементы с нечетными номерами (новые элементы

добавлять после существующих элементов с такими же значениями) и вывести ссылку на последний элемент преобразованного списка.

Dynamic39. Дан первый элемент A_1 непустого двусвязного списка. Продублировать в списке все элементы с нечетными значениями (новые элементы добавлять перед существующими элементами с такими же значениями) и вывести ссылку на первый элемент преобразованного списка.

Dynamic40. Дан первый элемент A_1 непустого двусвязного списка. Продублировать в списке все элементы с нечетными значениями (новые элементы добавлять после существующих элементов с такими же значениями) и вывести ссылку на последний элемент преобразованного списка.

Dynamic41. Дана ссылка A_0 на один из элементов непустого двусвязного списка. Удалить из списка данный элемент и вывести две ссылки: на элемент, предшествующий удаленному, и на элемент, следующий за удаленным (один или оба этих элемента могут отсутствовать; для отсутствующих элементов выводить null). После удаления элемента из списка вызвать для него метод Dispose.

Dynamic42. Дан первый элемент A_1 двусвязного списка, содержащего не менее двух элементов. Удалить из списка все элементы с нечетными номерами и вывести ссылку на первый элемент преобразованного списка. После удаления элементов из списка вызывать для них метод Dispose.

Dynamic43. Дан первый элемент A_1 непустого двусвязного списка. Удалить из списка все элементы с нечетными значениями и вывести ссылку на первый элемент преобразованного списка (если в результате удаления элементов список окажется пустым, то вывести null). После удаления элементов из списка вызывать для них метод Dispose.

Dynamic44. Дана ссылка A_0 на один из элементов непустого двусвязного списка. Переместить данный элемент в конец списка и вывести ссылки на первый и последний элементы преобразованного списка. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic45. Дана ссылка A_0 на один из элементов непустого двусвязного списка. Переместить данный элемент в начало списка и вывести ссылки на первый и последний элементы преобразованного списка. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic46. Дано число K (>0) и ссылка A_0 на один из элементов непустого двусвязного списка. Переместить в списке данный элемент на K позиций вперед (если после данного элемента находится менее K элементов, то

переместить его в конец списка). Вывести ссылки на первый и последний элементы преобразованного списка. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic47. Дано число K (>0) и ссылка A_0 на один из элементов непустого двусвязного списка. Переместить в списке данный элемент на K позиций назад (если перед данным элементом находится менее K элементов, то переместить его в начало списка). Вывести ссылки на первый и последний элементы преобразованного списка. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic48. Даны ссылки A_X и A_Y на два различных элемента двусвязного списка (элемент A_X находится в списке перед элементом A_Y, но не обязательно рядом с ним). Поменять местами данные элементы и вывести ссылку на первый элемент преобразованного списка. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic49°. Дан первый элемент A_1 непустого двусвязного списка. Перегруппировать элементы списка, переместив все элементы с нечетными номерами в конец списка (в том же порядке) и вывести ссылку на первый элемент преобразованного списка. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic50. Дан первый элемент A_1 непустого двусвязного списка. Перегруппировать элементы списка, переместив все элементы с нечетными значениями в конец списка (в том же порядке) и вывести ссылку на первый элемент преобразованного списка. Новые объекты типа Node не создавать, свойства Data не изменять.

Dynamic51. Для двух непустых двусвязных списков даны следующие объекты: A_1 и A_2 — начало и конец первого списка, A_0 — один из элементов второго списка. Объединить исходные списки, поместив все элементы первого списка (в том же порядке) перед данным элементом второго списка, и вывести ссылки на первый и последний элементы объединенного списка. Новые объекты типа Node не создавать.

Dynamic52. Для двух непустых двусвязных списков даны следующие объекты: A_1 и A_2 — начало и конец первого списка, A_0 — один из элементов второго списка. Объединить исходные списки, поместив все элементы первого списка (в том же порядке) после данного элемента второго списка, и вывести ссылки на первый и последний элементы объединенного списка. Новые объекты типа Node не создавать.

Dynamic53. Даны ссылки A_X и A_Y на два различных элемента двусвязного списка; элемент A_X находится в списке перед элементом A_Y, но не обязательно рядом с ним. Переместить элементы, расположенные между данными элементами (включая данные элементы), в новый список (в том же порядке). Вывести ссылки на первые элементы преобразованного и нового списков. Если преобразованный список окажется пустым, то связанную с ним ссылку положить равной null. Новые объекты типа Node не создавать.

Dynamic54. Даны ссылки A_X и A_Y на два различных элемента двусвязного списка; элемент A_X находится в списке перед элементом A_Y, но не обязательно рядом с ним. Переместить элементы, расположенные между данными элементами (не включая данные элементы), в новый список (в том же порядке). Вывести ссылки на первые элементы преобразованного и нового списков. Если новый список окажется пустым, то связанную с ним ссылку положить равной null. Новые объекты типа Node не создавать.

Dynamic55°. Дан первый элемент A_1 непустого двусвязного списка. Преобразовать список в *циклический*, записав в свойство Next последнего элемента списка ссылку на его первый элемент, а в свойство Prev первого элемента — ссылку на последний элемент. Вывести ссылку на элемент, который был последним элементом исходного списка.

Dynamic56. Даны ссылки A_1 и A_2 на первый и последний элементы непустого двусвязного списка, содержащего четное количество элементов. Преобразовать список в два *циклических* списка (см. задание Dynamic55), первый из которых содержит первую половину элементов исходного списка, а второй — вторую половину. Вывести ссылки A_3 и A_4 на два средних элемента исходного списка (элемент A_3 должен входить в первый циклический список, а элемент A_4 — во второй). Новые объекты типа Node не создавать.

Dynamic57. Дано число K (>0) и ссылки A_1 и A_2 на первый и последний элементы непустого двусвязного списка. Осуществить *циклический сдвиг* элементов списка на K позиций вперед (то есть в направлении от начала к концу списка) и вывести ссылки на первый и последний элементы полученного списка. Для выполнения циклического сдвига преобразовать исходный список в циклический (см. задание Dynamic55), после чего «разорвать» его в позиции, соответствующей данному значению K. Новые объекты типа Node не создавать.

Dynamic58. Дано число K (>0) и ссылки A_1 и A_2 на первый и последний

элементы непустого двусвязного списка. Осуществить *циклический сдвиг* элементов списка на K позиций назад (то есть в направлении от конца к началу списка) и вывести ссылки на первый и последний элементы полученного списка. Для выполнения циклического сдвига преобразовать исходный список в циклический (см. задание Dynamic55), после чего «разорвать» его в позиции, соответствующей данному значению K. Новые объекты типа Node не создавать.

Dynamic59°. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы двусвязного списка (если список является пустым, то $A_1 = A_2 = A_3 =$ null). Также дано число N (>0) и набор из N чисел. Описать класс IntList, содержащий следующие члены:

- три закрытых поля first, last, current типа Node (*первый*, *последний* и *текущий* элементы списка);
- конструктор с параметрами aFirst, aLast, aCurrent — первым, последним и текущим элементами существующего списка;
- процедура InsertLast(D), которая добавляет новый элемент со значением D в конец списка (D — входной параметр целого типа, добавленный элемент становится текущим);
- процедура Put (без параметров), которая выводит ссылки на поля first, last и current, используя метод Put класса PT.

С помощью метода InsertLast добавить в конец исходного списка данный набор чисел (в том же порядке) и вывести ссылки на первый, последний и текущий элементы полученного списка, используя для этого метод Put класса IntList.

Dynamic60. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы двусвязного списка (если список является пустым, то $A_1 = A_2 = A_3 =$ null). Также дано число N (>0) и набор из N чисел. Включить в класс IntList (см. задание Dynamic59) процедуру InsertFirst(D), которая добавляет новый элемент со значением D в начало списка (D — входной параметр целого типа). Добавленный элемент становится текущим. С помощью метода InsertFirst добавить в начало исходного списка данный набор чисел (добавленные числа будут располагаться в списке в обратном порядке) и вывести ссылки на первый, последний и текущий элементы полученного списка.

Dynamic61. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы непустого двусвязного списка. Также даны пять чисел. Включить в

класс IntList (см. задание Dynamic59) процедуру InsertBefore(D), которая вставляет новый элемент со значением D перед текущим элементом списка (D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью метода InsertBefore вставить пять данных чисел в исходный список и вывести ссылки на первый, последний и текущий элементы полученного списка.

Dynamic62. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы непустого двусвязного списка. Также даны пять чисел. Включить в класс IntList (см. задание Dynamic59) процедуру InsertAfter(D), которая вставляет новый элемент со значением D после текущего элемента списка (D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью метода InsertAfter вставить пять данных чисел в исходный список и вывести ссылки на первый, последний и текущий элементы полученного списка.

Dynamic63°. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы непустого двусвязного списка. Включить в класс IntList (см. задание Dynamic59) процедуры ToFirst (делает текущим первый элемент списка), ToNext (делает текущим в списке следующий элемент, если он существует), SetData(D) (присваивает текущему элементу списка значение D целого типа), а также функцию IsLast логического типа (возвращает TRUE, если текущий элемент списка является его последним элементом, и FALSE в противном случае). Методы ToFirst, ToNext и IsLast не имеют параметров; параметр D метода SetData является входным параметром целого типа. С помощью данных методов присвоить нулевые значения элементам исходного списка с нечетными номерами и вывести количество элементов в списке, а также ссылки на первый, последний и текущий элементы преобразованного списка.

Dynamic64. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы непустого двусвязного списка. Включить в класс IntList (см. задание Dynamic59) процедуры ToLast (делает текущим последний элемент списка), ToPrev (делает текущим в списке предыдущий элемент, если он существует) и функции GetData целого типа (возвращает значение текущего элемента списка), IsFirst логического типа (возвращает TRUE, если текущий элемент списка является его первым элементом, и FALSE в противном случае). Данные методы не имеют параметров. С помощью этих методов вывести все четные значения элементов исходного списка, просматривая

список с конца. Вывести также количество элементов в списке.

Dynamic65. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы двусвязного списка, содержащего не менее пяти элементов. Включить в класс IntList (см. задание Dynamic59) функцию DeleteCurrent целого типа (без параметров), удаляющую из списка текущий элемент и возвращающую его значение. После удаления элемента текущим становится следующий элемент или, если следующего элемента не существует, последний элемент списка. Метод DeleteCurrent также вызывает для удаленного элемента метод Dispose. С помощью метода DeleteCurrent удалить из исходного списка пять элементов и вывести их значения. Вывести также ссылки на первый, последний и текущий элементы преобразованного списка (для пустого списка вывести три константы null).

Dynamic66. Даны ссылки A_1, A_2 и A_3 на первый, последний и текущий элементы непустого двусвязного списка. Включить в класс IntList (см. задание Dynamic59) классовый метод — процедуру Split(L_1, L_2), которая переносит элементы списка L_1 от текущего до последнего в новый список L_2 (таким образом, список L_1 делится на две части, причем первая часть может оказаться пустой). Параметры процедуры имеют тип IntList; первый параметр является входным, второй — выходным. Текущими элементами непустых результирующих списков становятся их первые элементы. Новые объекты типа Node в процедуре Split не создавать. С помощью этой процедуры разбить исходный список на два и вывести ссылки на первый, последний и текущий элементы каждого из полученных списков (для пустого списка выводятся три константы null).

Dynamic67. Даны ссылки на первый, последний и текущий элементы двух непустых двусвязных списков. Включить в класс IntList (см. задание Dynamic59) классовый метод — процедуру Add(L_1, L_2), которая добавляет все элементы из списка L_1 (в том же порядке) в конец списка L_2; в результате список L_1 становится пустым. Текущим элементом дополненного списка становится первый из добавленных элементов. Оба параметра процедуры имеют тип IntList и являются входными. Новые объекты типа Node в процедуре Add не создавать. С помощью этой процедуры добавить первый из данных списков в конец второго и вывести ссылки на первый, последний и текущий элементы объединенного списка.

Dynamic68. Даны ссылки на первый, последний и текущий элементы двух непустых двусвязных списков. Включить в класс IntList (см. задание

Dynamic59) классовый метод — процедуру Insert(L_1, L_2), которая вставляет все элементы из списка L_1 (в том же порядке) в список L_2 перед его текущим элементом; в результате список L_1 становится пустым. Текущим элементом списка L_2 становится первый из вставленных элементов. Оба параметра процедуры имеют тип IntList и являются входными. Новые объекты типа Node в процедуре Insert не создавать. С помощью этой процедуры вставить первый из данных списков в текущую позицию второго и вывести ссылки на первый, последний и текущий элементы объединенного списка.

Dynamic69. Даны ссылки на первый, последний и текущий элементы двух двусвязных списков (второй список может быть пустым). Включить в класс IntList (см. задание Dynamic59) классовый метод — процедуру MoveCurrent(L_1, L_2), которая перемещает текущий элемент списка L_1 в список L_2 (элемент вставляется после текущего элемента списка L_2 и сам становится текущим; в списке L_1 текущим становится следующий элемент или, если следующего элемента не существует, последний элемент). Оба параметра процедуры имеют тип IntList и являются входными. Новые объекты типа Node в процедуре MoveCurrent не создавать. С помощью этой процедуры переместить текущий элемент первого списка во второй и вывести ссылки на первый, последний и текущий элементы каждого из полученных списков (для пустого списка выводятся три константы null).

Список с барьерным элементом

Использованная в заданиях Dynamic31–Dynamic69 реализация двусвязного списка в виде цепочки узлов, ограниченной по краям нулевыми ссылками null, не является единственно возможной. Двусвязный список можно также реализовать в виде *замкнутой* цепочки узлов с дополнительным *фиктивным*, или *барьерным*, элементом. Этот барьерный элемент связан своими свойствами Next и Prev с первым и последним «настоящим» элементом списка соответственно, поэтому, имея ссылку на барьерный элемент, можно сразу перейти как к первому, так и к последнему элементу списка (естественно, первый и последний элементы также связаны с барьерным элементом своими свойствами Prev и Next соответственно). Для работы с двусвязным списком, снабженным барьерным элементом, достаточно хранить две ссылки: Barrier, указывающую на барьерный элемент, и Current, указывающую на текущий элемент (который может быть как «настоящим», так и барьерным элементом). Свойство Data ба-

рьерного элемента может быть произвольным; для определенности его можно положить равным 0. *Пустой список* в данной реализации представляет собой единственный барьерный элемент, «замкнутый на себя».

Задания Dynamic70–Dynamic80 посвящены *двусвязным спискам с барьерным элементом*.

Dynamic70°. Даны ссылки A_1 и A_2 на первый и последний элементы двусвязного списка, реализованного в виде цепочки узлов, которая ограничена по краям константами null (если список пуст, то $A_1 = A_2 =$ null). Преобразовать исходный список в *циклический список* (см. задание Dynamic55), снабженный *барьерным элементом*. Барьерный элемент должен иметь значение 0 и быть связан своими свойствами Next и Prev с первым и последним элементом исходного списка (в случае пустого исходного списка свойства Next и Prev барьерного элемента должны указывать на сам барьерный элемент). Вывести ссылку на барьерный элемент полученного списка. Не создавать новые объекты типа Node, за исключением барьерного элемента.

Dynamic71. Даны ссылки A_1 и A_2 на барьерный и текущий элементы двусвязного списка (о списке с барьерным элементом см. задание Dynamic70). Разбить список на два, перенеся во второй список все элементы от текущего до последнего и добавив ко второму списку барьерный элемент. Если текущий элемент исходного списка является барьерным элементом, то второй список должен быть пустым (то есть состоять только из барьерного элемента). Вывести ссылку на барьерный элемент второго списка. Не создавать новые объекты типа Node, за исключением барьерного элемента для второго списка.

Dynamic72. Даны ссылки A_1 и A_2 на барьерные элементы двух двусвязных списков (о списке с барьерным элементом см. задание Dynamic70). Объединить исходные списки, связав конец первого и начало второго списка (барьерным элементом объединенного списка должен остаться барьерный элемент первого списка). Вывести ссылки на первый и последний элементы объединенного списка (если объединенный список является пустым, то дважды вывести ссылку на его барьерный элемент). После удаления лишнего барьерного элемента вызвать для него метод Dispose.

Dynamic73. Даны ссылки A_1 и A_2 на барьерные элементы двух двусвязных списков (о списке с барьерным элементом см. задание Dynamic70). Объединить исходные списки, связав конец первого и начало второго списка

(барьерным элементом объединенного списка должен остаться барьерный элемент второго списка). Вывести ссылки на первый и последний элементы объединенного списка (если объединенный список является пустым, то дважды вывести ссылку на его барьерный элемент). После удаления лишнего барьерного элемента вызвать для него метод Dispose.

Dynamic74°. Даны ссылки A_1 и A_2 на барьерный и текущий элементы двусвязного списка (о списке с барьерным элементом см. задание Dynamic70). Также дано число N (>0) и набор из N чисел. Описать класс IntListB, содержащий следующие члены:

- закрытые поля barrier и current типа Node (*барьерный* и *текущий* элементы списка);
- конструктор с параметрами aBarrier и aCurrent — барьерным и текущим элементами существующего списка;
- процедура InsertLast(D), которая добавляет новый элемент со значением D в конец списка (D — входной параметр целого типа, добавленный элемент становится текущим);
- процедура Put (без параметров), которая выводит ссылку на поле current, используя метод Put класса PT.

С помощью метода InsertLast добавить в конец исходного списка данный набор чисел (в том же порядке) и вывести сссылку на текущий элемент полученного списка, используя для этого метод Put класса IntListB.

Dynamic75. Даны ссылки A_1 и A_2 на барьерный и текущий элементы двусвязного списка. Также дано число N (>0) и набор из N чисел. Включить в класс IntListB (см. задание Dynamic74) процедуру InsertFirst(D), которая добавляет новый элемент со значением D в начало списка (D — входной параметр целого типа). Добавленный элемент становится текущим. С помощью метода InsertFirst добавить в начало исходного списка данный набор чисел (добавленные числа будут располагаться в списке в обратном порядке) и вывести ссылку на текущий элемент полученного списка.

Dynamic76. Даны ссылки A_1 и A_2 на барьерный и текущий элементы двусвязного списка. Также даны пять чисел. Включить в класс IntListB (см. задание Dynamic74) процедуру InsertBefore(D), которая вставляет новый элемент со значением D перед текущим элементом списка (D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью метода InsertBefore вставить пять данных чисел в исходный список и вывести ссылку на текущий элемент полученного списка.

Dynamic77. Даны ссылки A_1 и A_2 на барьерный и текущий элементы двусвязного списка. Также даны пять чисел. Включить в класс IntListB (см. задание Dynamic74) процедуру InsertAfter(D), которая вставляет новый элемент со значением D после текущего элемента списка (D — входной параметр целого типа). Вставленный элемент становится текущим. С помощью метода InsertAfter вставить пять данных чисел в исходный список и вывести ссылку на текущий элемент полученного списка.

Dynamic78°. Даны ссылки A_1 и A_2 на барьерный и текущий элементы двусвязного списка. Включить в класс IntListB (см. задание Dynamic74) процедуры ToFirst (делает текущим первый элемент списка), ToNext (делает текущим следующий элемент в списке), SetData(D) (присваивает текущему элементу списка значение D целого типа, если данный элемент не является барьерным) и функцию IsBarrier логического типа (возвращает TRUE, если текущий элемент списка является его барьерным элементом, и FALSE в противном случае). Методы ToFirst, ToNext и IsBarrier не имеют параметров. Параметр D метода SetData является входным параметром целого типа. С помощью этих методов присвоить нулевые значения элементам исходного списка с нечетными номерами и вывести количество элементов в списке, а также ссылку на новый текущий элемент списка. Нумерация ведется от первого элемента списка; барьерный элемент не нумеруется и не учитывается при подсчете элементов.

Dynamic79. Даны ссылки A_1 и A_2 на барьерный и текущий элементы двусвязного списка. Включить в класс IntListB (см. задание Dynamic74) процедуры ToLast (делает текущим последний элемент списка), ToPrev (делает текущим предыдущий элемент в списке) и функцию GetData целого типа (возвращает значение текущего элемента списка L). Данные методы не имеют параметров. С помощью этих методов, а также с использованием функции IsBarrier из задания Dynamic78, вывести все четные значения элементов исходного списка, просматривая список с конца. Вывести также количество элементов в списке. Барьерный элемент не обрабатывается и не учитывается при подсчете элементов.

Dynamic80. Даны ссылки A_1 и A_2 на барьерный и текущий элементы непустого двусвязного списка, причем текущий элемент не совпадает с барьерным. Включить в класс IntListB (см. задание Dynamic74) функцию DeleteCurrent целого типа, удаляющую из списка текущий элемент и возвращающую его значение. Текущим становится следующий элемент или,

если следующий элемент является барьерным, предыдущий элемент списка. Функция также вызывает для удаленного элемента метод Dispose. Если текущим элементом является барьерный элемент, то функция не выполняет никаких действий и возвращает 0. С помощью этой функции, а также метода IsBarrier из задания Dynamic78, удалить из исходного списка пять элементов (или все элементы, если их менее пяти) и вывести их значения. Вывести также ссылку на новый текущий элемент списка.

Литература

1. *Абрамян М. Э.* 1000 задач по программированию. Часть I: Скалярные типы данных, управляющие операторы, процедуры и функции. — УПЛ РГУ, 2004. — 43 с.
2. *Абрамян М. Э.* 1000 задач по программированию. Часть II: Минимумы и максимумы, одномерные и двумерные массивы, символы и строки, двоичные файлы. — УПЛ РГУ, 2004. — 42 с.
3. *Абрамян М. Э.* 1000 задач по программированию. Часть III: Текстовые файлы, составные типы данных в процедурах и функциях, рекурсия, указатели и динамические структуры. — УПЛ РГУ, 2004. — 43 с.
4. *Абрамян М. Э., Михалкович С. С.* Основы программирования на языке Паскаль: Скалярные типы данных, управляющие операторы, процедуры и функции. 2-е изд. — Ростов-на-Дону: «ЦВВР», 2005. — 198 с.
5. *Абрамян М. Э.* Практикум по программированию на языке Паскаль: Массивы, строки, файлы, рекурсия, динамические структуры. 5-е изд., перераб. — Ростов-на-Дону: «ЦВВР», 2006. — 187 с.
6. *Абрамян М. Э.* Практикум по программированию на языках C# и VB .NET. — Ростов-на-Дону: «ЦВВР», 2006. — 220 с.

Содержание