Нумерация: год-вариант-вопрос. Сортировка: вопрос, год, вариант.

Оглавление

[Вопрос №1 3](#_Toc534456809)

[2015-1-1 3](#_Toc534456810)

[2015-2-1 3](#_Toc534456811)

[2016-1-1 3](#_Toc534456812)

[2016-2-1 3](#_Toc534456813)

[2017-1-1 3](#_Toc534456814)

[2017-2-1 3](#_Toc534456815)

[2018-1-1 3](#_Toc534456816)

[2018-2-1 4](#_Toc534456817)

[Вопрос №2 5](#_Toc534456818)

[2015-1-2 5](#_Toc534456819)

[2015-2-2 5](#_Toc534456820)

[2016-1-2 5](#_Toc534456821)

[2016-2-2 5](#_Toc534456822)

[2017-1-2 5](#_Toc534456823)

[2017-2-2 5](#_Toc534456824)

[2018-1-2 6](#_Toc534456825)

[2018-2-2 6](#_Toc534456826)

[Вопрос №3 7](#_Toc534456827)

[2015-1-3 7](#_Toc534456828)

[2015-2-3 7](#_Toc534456829)

[2016-1-3 7](#_Toc534456830)

[2016-2-3 7](#_Toc534456831)

[2017-1-3 7](#_Toc534456832)

[2017-2-3 7](#_Toc534456833)

[2018-1-3 7](#_Toc534456834)

[2018-2-3 7](#_Toc534456835)

[Вопрос №4 8](#_Toc534456836)

[2015-1-4 8](#_Toc534456837)

[2015-2-4 8](#_Toc534456838)

[2016-1-4 8](#_Toc534456839)

[2016-2-4 8](#_Toc534456840)

[2017-1-4 8](#_Toc534456841)

[2017-2-4 8](#_Toc534456842)

[2018-1-4 9](#_Toc534456843)

[2018-2-4 9](#_Toc534456844)

[Вопрос №5 10](#_Toc534456845)

[2015-1-5 10](#_Toc534456846)

[2015-2-5 10](#_Toc534456847)

[2016-1-5 10](#_Toc534456848)

[2016-2-5 10](#_Toc534456849)

[2017-1-5 10](#_Toc534456850)

[2017-2-5 10](#_Toc534456851)

[2018-1-5 10](#_Toc534456852)

[2018-2-5 10](#_Toc534456853)

[Вопрос №6 11](#_Toc534456854)

[2015-1-6 11](#_Toc534456855)

[2015-2-6 11](#_Toc534456856)

[2016-1-6 11](#_Toc534456857)

[2016-2-6 11](#_Toc534456858)

[2017-1-6 11](#_Toc534456859)

[2017-2-6 11](#_Toc534456860)

[2018-1-6 11](#_Toc534456861)

[2018-2-6 11](#_Toc534456862)

[Вопрос №7 12](#_Toc534456863)

[2015-1-7 12](#_Toc534456864)

[2015-2-7 12](#_Toc534456865)

[2016-1-7 12](#_Toc534456866)

[2016-2-7 12](#_Toc534456867)

[2017-1-7 12](#_Toc534456868)

[2017-2-7 12](#_Toc534456869)

[2018-1-7 12](#_Toc534456870)

[2018-2-7 12](#_Toc534456871)

[Вопрос №8 13](#_Toc534456872)

[2015-1-8 (1) 13](#_Toc534456873)

[2015-1-8 (2) 13](#_Toc534456874)

[2015-2-8 (1) 14](#_Toc534456875)

[2015-2-8 (1) 14](#_Toc534456876)

[2016-1-8 15](#_Toc534456877)

[2016-2-8 16](#_Toc534456878)

[2017-1-8 16](#_Toc534456879)

[2017-2-8 16](#_Toc534456880)

[2018-1-8 17](#_Toc534456881)

[2018-1-8 17](#_Toc534456882)

# Вопрос №1

## 2015-1-1

Что означает операция >>> в языке Java? Почему её нет в языках С/C++, C#?

## 2015-2-1

В каких из языков: Си++, Модула-2, Java, Delphi, C#, Оберон, Python, JavaScript — можно реагировать на ошибки переполнения, связанные с целочисленной арифметикой? Для каждого языка приведите пример соответствующих конструкций.

## 2016-1-1

Объясните, что означает понятие «виртуальное множественное наследование» в языке С++. Объясните, какие проблемы возникают при реализации этого понятия? Не перечисляйте общих для множественного наследования проблем, а ограничьтесь только специфическими проблемами виртуального наследования.

## 2016-2-1

Объясните, что означает понятие «множественное наследование» в языке С++. Объясните, какие проблемы возникают при реализации виртуальных функций при множественном наследовании (не перечисляйте общих для множественного наследования проблем, а ограничьтесь только специфическими проблемами реализации виртуальных методов при множественном наследовании)?

## 2017-1-1

Есть ли ошибка в тексте приведенной ниже функции r\_vowels(s) на языке JavaScript (функция подсчитывает число гласных букв русского алфавита в строке-аргументе s)? Если есть, то объясните, в чем она состоит и как изменить текст функции, чтобы она работала правильно во всех реализациях языка?

**function** r\_vowels(s) {

**var** i,cnt = **0**; **const** vowels = 'АЕЁИОУЫЭЮЯаеёиоуыэюя';

**for** (i=**0**; i<s.length; ++i) **if** (vowels.indexOf(s[i])>=**0**) ++cnt;

**return** cnt;

}

## 2017-2-1

Есть ли ошибка в тексте приведенной ниже функции r\_vowels(s) на языке C# (функция подсчитывает число гласных букв русского алфавита в строке-аргументе s)? Если есть, то объясните, в чем она состоит и как изменить текст функции, чтобы она работала правильно во всех реализациях языка?

**public** **static** **int** **r\_vowels**(**string** s) {

**int** i,cnt = **0**; **const** **string** vowels = "АЕЁИОУЫЭЮЯаеёиоуыэюя";

**for** (i=**0**; i<s.Length; ++i) **if** (vowels.IndexOf(s[i])>=**0**) ++cnt;

**return** cnt;

}

## 2018-1-1

Замените знаки вопроса (?????) в приведенной ниже программе на языке С# так, чтобы в стандартный вывод выдавалась сумма квадратов элементов списка l. Никакой код куда- либо еще добавлять нельзя. Примечание: метод ForEach списка имеет прототип List<**int**>.ForEach(Action<**int**> action) и выполняет действие action для каждого элемента списка.

List<**int**> l = **new** List<**int**>() { **1**, **2**, **3**, **5**, **7**, **9** };

**int** s = **0**;

l.ForEach(?????);

System.Console.WriteLine(s);

## 2018-2-1

Замените знаки вопроса (?????) в приведенном ниже фрагменте программе на языке С++ так, чтобы в стандартный вывод выдавалась сумма элементов массива a. Никакой код куда-либо еще добавлять нельзя.

**int** x = **0**; **int** a[] = { **1**,**3**,**5**,**7**,**9** }; **size\_t** n = **sizeof**(a)/**sizeof** a[**0**];

std::for\_each(a, a + n, ?????);

std::cout << x << std::endl;

# Вопрос №2

## 2015-1-2

Напишите спецификацию шаблона template <int N> struct Fun {....}; на языке С++ так, чтобы значение Fun<N>::result было равно сумме квадратов целых чисел от 1 до N для любого неотрицательного целого константного значения N.

## 2015-2-2

Напишите спецификацию шаблона template <int N> struct ArraySum {....}; на языке С++ так, чтобы для любого неотрицательного целого константного значения n значение ArraySum<n>::result было равно сумме первых n элементов массива Arr, описанного следующим образом:

static constexpr int Arr[] = {1,5,-3,.../\* и так далее \*/...};

## 2016-1-2

Объясните, что означают термины «семантика возобновления» и «семантика завершения». Приведите примеры языков, в которых реализована семантика завершения. Опишите пример моделирования семантики возобновления через семантику завершения.

## 2016-2-2

Объясните, что означают термины «семантика возобновления» и «семантика завершения». Приведите пример языка, в котором реализована семантика возобновления. В каких случаях семантика возобновления может оказаться предпочтительнее, чем семантика завершения?

## 2017-1-2

Что будет выдано в стандартный канал вывода после выполнения следующей программы на С++? Считать, что все нужные директивы include и using добавлены в текст.

**class** **TT** {

**char** \* \_p;

**public:**

TT(**const** TT& t) { \_p = strdup(t.\_p); cout << "TT& " << \_p<< endl;}

**explicit** TT(**const** **char** \* p = "") {\_p = strdup(p); cout << "Hello, "<< \_p << endl; }

TT& **operator** = (**const** TT& t) {

**if** (**this** != &t) {

cout << "goodbye, " << \_p << endl; free(\_p); \_p = strdup(t.\_p);

}

cout << "Hello, " << \_p << endl; **return** \***this**;

}

~TT() { cout << "goodbye, " << \_p << endl; free(\_p);}

};

**int** **main**() { TT t("world"); t = TT("bar"); **return** **0**; }

Добавьте в класс TT РОВНО ОДИН метод так, чтобы результатом стали следующие строки:

Hello, world

Hello, bar

Hello, bar

goodbye, world

goodbye, bar

## 2017-2-2

Что будет выдано в стандартный канал вывода после выполнения следующей программы на С++? Считать, что все нужные директивы include и using добавлены в текст.

**class** **TT** {

**int** \_p;

**public:**

TT(**const** TT& t) : \_p(t.\_p) { cout << "TT& " << \_p<< endl;}

**explicit** TT(**int** p = **0**) : \_p(p) { cout << "Hello, "<< \_p << endl; }

~TT() { cout << "goodbye, " << \_p << endl;}

};

**int** **main**() { list<TT> v; v.push\_back(TT(**1**)); v.push\_back(TT(**2**)); **return** **0**; }

Добавьте в класс TT РОВНО ОДИН метод так, чтобы результатом стали следующие строки (<CR> означает переход на новую строку):

Hello, 1<CR> TT&& 1<CR> goodbye, -1<CR> Hello, 2<CR> TT&& 2<CR> goodbye, -2<CR> goodbye, 1<CR> goodbye, 2<CR>

## 2018-1-2

Пусть s описана как переменная строкового типа (string), и в программе есть присваивание s= ”кафе” (все буквы - кириллические). Какова будет длина этой строки в языках C++ (s.size()), Java (s.length), Go (len(s)), Swift (s.count)? Для каждого языка дайте ответ и объясните различия (если они есть).

## 2018-2-2

Пусть s1 и s2 описаны как переменные строкового типа (string), им присвоены непустые (длина >= 1) строки.. Можно ли утверждать, что длина конкатенации строк (s = s1 + s2) будет равна сумме длин s1 и s2 для любого языка программирования? Варианты языков: C++ (s.size()), Java (s.length), Go (len(s)), Swift (s.count), C# (s.Length), Python (len(s)). В скобках для каждого языка указаны функции или свойства, возвращающие длину строки.

# Вопрос №3

## 2015-1-3

В каких из языков: Си++, Java, C#, Оберон, Оберон-2 — можно объявлять приватные абстрактные функции? Если можно, то для каждого языка приведите пример.

## 2015-2-3

Можно ли в языке С++ определить тело абстрактной функции? Если можно, то приведите пример контекста, в котором вызывается это тело.

## 2016-1-3

Дайте определение языковой конструкции «интерфейс». Что означает термин «интеграция интерфейсов в язык программирования»? Приведите два различных примера интерфейсов, интегрированных с языком Java.

## 2016-2-3

Дайте определение языковой конструкции «интерфейс». Что означает термин «интеграция интерфейсов в язык программирования»? Приведите два различных примера интерфейсов, интегрированных с языком С#.

## 2017-1-3

Объясните, что означает понятие «сопрограммы Кнута». В чем отличие сопрограмм языка Python от сопрограмм Кнута? Приведите пример языка, который ближе, чем Python, реализует сопрограммы Кнута (с примерами соответствующих языковых конструкций).

## 2017-2-3

Дайте определение функции-генератора языка Python. Напишите пример функции-генератора, включая ее использование. Как связаны понятия функции-генератора и сопрограммы в Python?

## 2018-1-3

Объясните, допустимо ли присваивание aobj = afoo в следующем фрагменте программы на Java. Если допустимо, то какие проблемы с aobj и afoo могут возникнуть в дальнейшем при выполнении программы?

**class** **Foo** { **public** **void** foo () {} }

**class** **Bar** { **public** **void** bar () {} }

Object [] aobj; Foo [] afoo = **new** Foo[**10**];

aobj = afoo;

## 2018-2-3

Объясните, допустимо ли присваивание x = y в следующем фрагменте программы на C#. Если допустимо, то какие проблемы с x и y могут возникнуть в дальнейшем при выполнении программы?

**class** **Foo** { **public** **void** foo () {} }

**class** **Bar**: Foo { **public** **void** bar () {} }

**object** [] x; Bar [] y = **new** Bar[**10**];

x = y;

# Вопрос №4

## 2015-1-4

Напишите определение класса Fib на языке C# так, чтобы следующий фрагмент программы выдавал первые 10 чисел Фибоначчи (Fib1 = Fib2 = 1, Fibn = Fibn-1 + Fibn-2):

foreach (var f in new Fib(100)) System.Console.WriteLine(f);

## 2015-2-4

Напишите определение функции-генератора с именем NN на языке Python так, чтобы следующий фрагмент программы выдавал первые 100 квадратов целых чисел (1,4,9,16,…):

**for** x **in** NN(**100**):

**print**(x)

## 2016-1-4

Дайте определение «пространства имен» в языке C# и пакета в языке Java. В чем состоит основное сходство и основное отличие этих понятий?

## 2016-2-4

Перечислите модули, которые могут быть единицами компиляции (ЕК) в языке Ада. Какие ЕК называются первичными, а какие — вторичными? В чем состоит отличие первичных и вторичных ЕК в этом языке?

## 2017-1-4

На языке Java написан класс Generator (не компилируется из-за отсутствия объявления имени и знаков вопроса в вызовах метода generate):

**public** **class** **Generator** {

**public** **static** **void** **main**(String[] args) {

generate(???, **10**); // заменить ??? на правильное значение

generate(???, **10**); // заменить ??? на правильное значение

}

**static** **void** **generate**(Compute func, **int** n) {

**for** (**int** i = **1**; i <= n; ++i) {

System.out.print(func.compute(i));

System.out.print(' ');

}

}

}

Объявить тип данных Compute и заменить вопросы в вызовах метода generate на правильные лямбда-выражения так, чтобы в канал вывода было выдано:

1 4 9 16 25 36 49 64 81 100 1 8 27 64 125 216 343 512 729 1000

## 2017-2-4

На языке Java написан класс CSV (не компилируется из-за отсутствия объявления имени Printer и знаков вопроса в вызовах метода print):

**public** **class** **CSV** {

**public** **static** **void** **main**(String[] args) {

**for** (**int** i = **0**; i < **10**; ++i)

print(???, i); // подставить правильное значение

print(???, **10**); // подставить правильное значение

}

**static** **void** **print**(Printer p, Object x) {

p.print(x);

}

**static** **void** **printInner**(Object x) {

System.out.print('\"');

System.out.print(x);

System.out.print("\", ");

}

**static** **void** **printLast**(Object x) {

System.out.print('\"');

System.out.print(x);

System.out.println("\"");

}

}

Объявить тип данных Printer и заменить вопросы в вызовах метода print на правильные значения (НЕ ИСПОЛЬЗУЯ лямбда-выражения) так, чтобы в канал вывода было выдано:

"0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10"

## 2018-1-4

Написать на языке С++ шаблонную функцию int MakeWord(), которая возвращает целое число, являющееся побитовой суммой (операция суммирования — побитовое «или» | ) аргументов шаблона. Аргументы шаблона должны быть целочисленными. Например std::cout << MakeWord<1,2>(); должно выдать 3, а std::cout << MakeWord<2,4,9>(); должно выдать 15.

## 2018-2-4

Написать на языке С++ шаблонную функцию MakeWord c переменным числом параметров (как своих, так и шаблонных), которая возвращает целое число, являющееся побитовой суммой (операция суммирования — побитовое «или» | ) своих параметров. Аргументы шаблона должны быть типами. Например std::cout << MakeWord(1,2); должно выдать 3, а std::cout << MakeWord(2,4,9); должно выдать 15.

# Вопрос №5

## 2015-1-5

Дайте общее определение операции индексирования в языке программирования. Напишите пример пользовательского переопределения операции индексирования в языке С++ (определение операции и ее вызов). Есть ли аналогичные средства в языке C#? Если есть, то приведите пример на этом языке, аналогичный приведенному для С++.

## 2015-2-5

Дайте общее определение операции преобразования типа в языке программирования. Напишите пример пользовательского переопределения операции преобразования типа в языке С++ (определение операции и ее вызов). Есть ли аналогичные средства в языке C#? Если есть, то приведите пример на этом языке, аналогичный приведенному для С++.

## 2016-1-5

Что будет напечатано в результате работы следующей программы на языке Python?

s = range(**1**,**5**)

sqr = True

processFunc = sqr **and** (**lambda** s: s\*s) **or** (**lambda** s: s)

**print**([processFunc(i) **for** i **in** s])

Что будет напечатано, если sqr = True заменить на sqr = False?

## 2016-2-5

Что будет напечатано в результате работы следующей программы на Javascript? Считаем, что метод console.log печатает значение аргумента на экран.

u = (**function** f() { **var** k = **0**; **return** **function**(x) { **return** x+k++;} }) ()

**for** (i = **0**; i < **5**; i++) console.log(u(i))

**for** (i--; i >= **0**; i--) console.log(u(i))

## 2017-1-5

Переписать программу из задачи 2017-1-4 на языке С# так, чтобы она выдавала в стандартный вывод те же самые значения (набор и сигнатуры методов класса Generator должны остаться без изменения). Для вывода использовать System.Console.Write(...).

## 2017-2-5

Переписать программу из задачи 2017-2-4 на языке С# так, чтобы она выдавала в стандартный канал вывода те же самые значения (набор и сигнатуры методов класса CSV должны остаться без изменения). Для вывода использовать System.Console.Write(...) и WriteLine(...).

## 2018-1-5

Являются ли реализации обобщенных классов на языке Java более эффективными, чем их необобщенные аналоги. Например, будет ли реализация обобщенного класса Stack<Integer> более эффективной, чем реализация класса Stack из Object?

## 2018-2-5

Являются ли реализации обобщенных классов на языке C# более эффективными, чем их необобщенные аналоги. Например, будет ли реализация обобщенного класса Stack<int> более эффективной, чем реализация класса Stack из Object?

# Вопрос №6

## 2015-1-6

Дайте определение и пример использования анонимной (лямбда-функции) в языке Python. В каких из языков: JavaScript, Ада, Си++, Оберон, Оберон-2 — есть аналогичное понятие? Для каждого такого языка приведите пример.

## 2015-2-6

Дайте определение и пример использования анонимной (лямбда-функции) в языке C#. В каких из языков: JavaScript, Ада, Си++, Оберон, Оберон-2 — есть аналогичное понятие? Для каждого такого языка приведите пример.

## 2016-1-6

Переписать программу из задачи 2016-1-5 на языке С# с использованием механизма лямбда- функций и обобщенных делегатов так, чтобы она выдавала в стандартный вывод те же самые значения.

## 2016-2-6

Переписать программу из задачи 2016-2-5 на языке С++ с использованием механизма лямбда- функций так, чтобы она выдавала в стандартный вывод те же самые значения.

## 2017-1-6

Объявление шаблонного класса CalcSort на языке С++ предполагает, что над параметром шаблона T выполняются 4 арифметические операции (+,-,\*,/) и операции сравнения:

template <typename T> class CalcSort { ... };

Напишите эквивалентное объявление обобщенного класса CalcSort на языке Java (опустив, как и выше, все объявления членов класса), добавив при необходимости нужные объявления типов вне этого класса.

## 2017-2-6

Объявление шаблонного класса CalcSort на языке С++ предполагает, что над параметром шаблона T выполняются 4 арифметические операции (+,-,\*,/) и операции сравнения:

template <typename T> class CalcSort { ... };

Напишите эквивалентное объявление обобщенного класса CalcSort на языке C# (опустив, как и выше, все объявления членов класса), добавив при необходимости нужные объявления типов вне этого класса.

## 2018-1-6

Напишите какой-нибудь пример генератора (но не сопрограммы) на языке Python. Чем сопрограммы языка Python отличаются от простых генераторов в этом языке?

## 2018-2-6

Напишите какой-нибудь пример сопрограммы на языке Python. Чем сопрограммы языка Python отличаются от go-программ языка Go?

# Вопрос №7

## 2015-1-7

Дайте определение понятия «защищенный член класса» в языке Java. Перечислите его отличия от аналогичного понятия в языке C#.

## 2015-2-7

Дайте определение понятия «объект, подобный массиву» в языке JavaScript. Перечислите его отличия от реального массива JavaScript.

## 2016-1-7

Рассмотрим файл на языке Java, содержащий определение класса С:

**package** test;

**class** **C** { **public** **void** f() { g(); } }

При трансляции этого файла выдается ошибка. Объясните, в чем она состоит, и что нужно добавить в программу, не меняя ничего в классе С, чтобы данный файл транслировался без ошибок? Можно добавлять в программу новые файлы, а также новые строки в заданный файл, но нельзя менять определение класса С.

## 2016-2-7

Рассмотрим фрагмент программы на языке Java, содержащий определение класса С:

**package** test;

**abstract** **class** **C** { **public** **abstract** **int** f(); }

**class** **D** {

**public** **void** **foo** () {

C c = **new** C {

**public** **int** **f**() { **return** **1**; }

};

}

}

При трансляции этого файла выдается ошибка. Объясните, из-за чего она выдается и что нужно добавить в программу, не меняя ничего в классе С, чтобы данный файл транслировался без ошибок? Можно только добавлять конструкции в программу, но не исключать.

## 2017-1-7

Дайте определение и пример локального анонимного класса в языке Java. Есть ли аналогичное понятие в языках С++ или С#?

## 2017-2-7

Дайте определение и пример внутреннего класса в языке Java. Есть ли аналогичное понятие в языках С++ или С#?

## 2018-1-7

Дайте определение вырезки из массива. Приведите примеры вырезок в языке Python и Go. Чем они отличаются?

## 2018-2-7

Дайте определение словаря (отображения, таблицы). В каких языках из списка : Python, C++, C# - словари входят в базис языка, а в каких — входят в стандартную библиотеку? Для каждого языка дайте пример (инициализации и обращения к элементу) словаря. Почему в JavaScript и PHP такого понятия нет ни в базисе, ни в стандартную библиотеке?

# Вопрос №8

## 2015-1-8 (1)

Что будет напечатано в результате работы следующей программы на С#?

**using** **System**;

**namespace** **test**

{

**class** **Program** {

**static** **void** **Main**() {

B a = **new** B(), b = **new** B(); P(a, b);

Console.WriteLine("------------------------------");

C c = **new** C(), d = **new** C(); P(c, d);

}

**static** **public** **void** **P**(A a, B b) {

a.f(); a.g();

b.f(); b.g();

}

}

**class** **A** {

**public** **void** **f**() { Console.WriteLine("A.f"); g(); }

**public** **virtual** **void** **g**() { Console.WriteLine("A.g"); }

}

**class** **B** : A {

**public** **virtual** **new** **void** **f**() { Console.WriteLine("B.f"); }

**public** **override** **void** **g**() { Console.WriteLine("B.g"); f(); }

}

**class** **C** : B {

**public** **override** **void** **f**() { Console.WriteLine("C.f"); }

**public** **override** **void** **g**() { Console.WriteLine("C.g"); f(); }

}

}

## 2015-1-8 (2)

Что будет напечатано в результате работы следующей программы на С#?

**using** **System**;

**namespace** **test** {

**class** **Program** {

**static** **void** **Main**() {

B a = **new** B(), b = **new** B(); P(a, b);

Console.WriteLine("------------------------------");

C c = **new** C(), d = **new** C(); P(c, d);

}

**static** **public** **void** **P**(A a, B b) {

a.f(); a.g();

b.f(); b.g();

}

}

**class** **A** {

**public** **void** **f**() { Console.WriteLine("A.f"); g(); }

**public** **virtual** **void** **g**() { Console.WriteLine("A.g"); }

}

**class** **B** : A {

**public** **void** **f**() { Console.WriteLine("B.f"); }

**public** **override** **void** **g**() { Console.WriteLine("B.g"); f(); }

}

**class** **C** : B {

**public** **void** **f**() { Console.WriteLine("C.f"); }

**public** **override** **void** **g**() { Console.WriteLine("C.g"); f(); }

}

}

## 2015-2-8 (1)

Что будет напечатано в результате работы следующей программы на Java?

**public** **class** **PLExamTest2014** {

**static** **void** **P**(A a, B b) {

a.g();

b.g();

}

**public** **static** **void** **main**(String[] args) {

P(**new** A(), **new** B());

System.out.println("------------------------------");

P(**new** B(), **new** C());

}

}

**class** **A** {

**private** **void** **pf**() {

System.out.println("A.pf");

}

**public** **void** **f**() {

System.out.println("A.f");

}

**public** **void** **g**() {

f();

pf();

}

}

**class** **B** **extends** A {

**public** **void** **pf**() {

System.out.println("B.pf");

}

**public** **void** **f**() {

System.out.println("B.f");

}

}

**class** **C** **extends** B {

**public** **void** **pf**() {

System.out.println("C.pf");

}

**public** **void** **f**() {

System.out.println("C.f");

}

}

## 2015-2-8 (1)

Что будет напечатано в результате работы следующей программы на Java?

**public** **class** **PLExamTest2014** {

**static** **void** **P**(A a, B b) {

a.g();

b.g();

}

**public** **static** **void** **main**(String[] args) {

P(**new** A(), **new** B());

System.out.println("------------------------------");

P(**new** B(), **new** C());

}

}

**class** **A** {

**private** **void** **pf**() {

System.out.println("A.pf");

}

**public** **void** **f**() {

System.out.println("A.f");

}

**public** **void** **g**() {

f();

pf();

}

}

**class** **B** **extends** A {

**private** **void** **pf**() {

System.out.println("B.pf");

}

**public** **void** **f**() {

System.out.println("B.f");

}

**public** **void** **g**() {

f();

pf();

}

}

**class** **C** **extends** B {

**public** **void** **pf**() {

System.out.println("C.pf");

}

**public** **void** **f**() {

System.out.println("C.f");

}

}

## 2016-1-8

Что будет напечатано в результате работы следующей программы на Java? Считаем, что

каждый пакет находится в своем файле.

**package** P2;

**class** **CC** **extends** P1.AC {

**public** **void** **f1**() { print("CC.f1");} **public** **void** f2() { print("CC.f2"); }

**public** **void** **f3**() { print("CC.f3");} **public** **void** f4() { print("CC.f4"); }

}

**public** **class** **TT** {

**public** **static** **void** **main**(String [] args) { **new** CC().show(); }

}

**package** P1;

**public** **abstract** **class** **AC** {

**public** **static** **final** **void** **print**(String s) { System.out.println(s); }

**public** **void** **f1**() { print("AC.f1"); }

**void** **f2**() { print("AC.f2"); }

**protected** **void** **f3**() { print("AC.f3"); }

**private** **void** **f4**() { print("AC.f4"); }

**public** **final** **void** **show**() { f1();f2();f3();f4(); }

}

## 2016-2-8

Что будет напечатано в результате работы следующей программы на Java?

**package** P1;

**public** **class** **T** {

**public** **static** **void** **main** (String [] args) { **new** CC().show(); }

}

**public** **abstract** **class** **AC** {

**public** **static** **final** **void** **print**(String s) { System.out.println(s); }

**public** **void** **f1**() { print("AC.f1"); }

**void** **f2**() { print("AC.f2"); }

**protected** **void** **f3**() { print("AC.f3"); }

**private** **void** **f4**() { print("AC.f4"); }

**public** **final** **void** **show**() { f1();f2();f3();f4(); }

}

**class** **CC** **extends** AC {

**public** **void** **f1**() { print("CC.f1"); }

**public** **void** **f2**() { print("CC.f2"); }

**public** **void** **f3**() { print("CC.f3"); }

**public** **void** **f4**() { print("CC.f4"); }

}

## 2017-1-8

Пусть программа на языке C# содержит следующий ошибочный фрагмент:

**class** **D** { }

**interface** IFace <T, R> {

R **Generate**();

**void** **Process**(T x);

}

IFace<D, Object> f1 = **new** ObjectFace(); // error!

IFace<D, Object> f2 = **new** DFace(); // error!

**class** **ObjectFace** : IFace<Object, Object> {

**public** Object **Generate**() { **return** **new** Object(); }

**public** **void** **Process**(Object x) {}

}

**class** **DFace** : IFace<D, D> {

**public** D **Generate**() { **return** **new** D(); }

**public** **void** **Process**(D x) { }

}

Объясните, в чем состоит ошибка, и исправьте РОВНО ОДНУ строчку во фрагменте так, чтобы ошибка исчезла.

## 2017-2-8

Пусть программа на языке Java содержит следующий ошибочный фрагмент:

**class** **D** { }

**interface** **IFace** <T, R>

{

R **Generate**();

**void** **Process**(T x);

**class** **ObjectFace** **implements** IFace<Object, Object> {

**public** Object **Generate**() { **return** **new** Object(); }

}

}

**public** **void** **Process**(Object x) {}

IFace<Object, D> i1;

i1 = **new** ObjectFace(); // error! i1 = new DFace(); //error!

**class** **DFace** **implements** IFace<D, D> {

**public** D **Generate**() { **return** **new** D(); }

}

**public** **void** **Process**(D x) { }

Объясните, в чем состоит ошибка, и исправьте РОВНО ОДНУ строчку во фрагменте так, чтобы ошибка исчезла.

## 2018-1-8

В приведенной ниже программе на языке Go есть ошибка, которая проявляется во время выполнения. В чем она заключается, и как ее исправить (указание — надо переставить одну строку в другое место)? Что будет выдано после исправления?

**package** main; **import** "fmt"

**func** fibonacci(c, quit **chan** **int**) {

x, y := **0**, **1**

**for** {

**select** {

**case** c <- x:

x, y = y, x+y

**case** <-quit:

fmt.Println("quit")

**return**

}

}

}

**func** main() {

c := make(**chan** **int**)

quit := make(**chan** **int**)

**go** **func**() {

**for** i := **0**; i < **10**; i++ {

fmt.Print(<-c);fmt.Print(" ")

}

}()

quit <- **0**

fibonacci(c, quit)

}

## 2018-1-8

В приведенной ниже программе на языке Go есть ошибка, которая проявляется во время выполнения. В чем она заключается, и как ее исправить (указание — надо закомментировать одну строку)? Что будет выдано после исправления?

**package** main;**import** "fmt"

**func** b(c **chan** **int**, quit **chan** **int**) {

**go** **func**() {

**for** i := **10**; i >= **0**; i-- {

c <- i

}

quit <- **0**

}()

}

**func** main() {

quit := make(**chan** **int**)

c := make(**chan** **int**)

b(c, quit)

**for** i := **10**; i >= **0**; i-- {

fmt.Print(<-c);fmt.Print(" ");

}

fmt.Print(<-quit)

**select** {

**case** c <- **1**: // do nothing

**case** <-quit:

**return**

}

}