МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. ЛОМОНОСОВА

Факультет вычислительной математики и кибернетики

ENGLISH READER IN COMPUTER SCIENCE

СБОРНИК СТАТЕЙ ПО ИТ НА АНГЛИЙСКОМ ЯЗЫКЕ

Авторы-составители:
Кашелкина О.А., Круглова М.А.

Под редакцией ***Кругловой М.А. и Саратовской Л.Б.***

Учебно-методическое пособие
для студентов второго курса

МОСКВА-2008

Авторы-составители: *Кашелкина О.А., Круглова М.А.*

Под редакцией *Кругловой М.А. и Саратовской Л.Б.*

Рецензенты: к.ф.н. Опарина О.И., [к.ф.н. Спиридонова Л.Ф.]

English Reader in Computer Science: Сб. ст. по ИТ на англ. яз.: Учебно-методическое пособие/ Авт.-сост.: Кашелкина О.А., Круглова М.А. Под ред. Кругловой М.А. и Саратовской. Л.Б. - М.: МАКС Пресс, 2008. - 128 с.
ISBN 978-5-89407-342-2
ISBN 978-5-317-02523-6

Данное учебно-методическое пособие предназначено для студентов второго курса факультета ВМиК и является продолжением аналогичного выпуска для студентов первого курса. Пособие состоит из 10 разделов, включающих оригинальные тексты из американских и английских профессиональных источников последних лет, и задания к ним.

УДК 378(076.8):811.111
ББК 81.2 Англ-923

Предисловие

Данное учебно-методическое пособие предназначено для студентов второго курса факультета ВМиК и является продолжением первого выпуска “English Reader for Computer Science” для студентов первого курса.

Материал пособия сгруппирован по темам, каждая из которых включает тексты и разработки к ним. Тексты пособия представляют собой оригинальные материалы, взятые из американских и английских профессиональных источников последних лет. Структура всех разделов учебного пособия однотипна: текст с предшествующим ему списком активной лексики и с последующими заданиями. Задания к текстам направлены на развитие у студентов навыков чтения и перевода, а также письменных и устных форм общения и дискуссии.

В конце пособия даны:

- список используемых сокращений

- и приложение “Appendix”, помогающее студентам второго курса научиться писать рефераты по своей специальности на английском языке.

Новым в учебно-методическом пособии для студентов второго курса сравнительно с первым изданием этой серии является также приложение “Revision Tests”, позволяющее проверить усвоение студентами лексического материала, представленного в текстах пособия.

От редактора

Contents

Database Management

Key vocabulary :

1. Database - база данных
2. Data redundancy - избыточность данных
3. Data integrity - целостность данных
4. Update (v.) - изменять, модифицировать
5. Storage (n.) - память, хранение информации
6. Indexing (n.) - индексация
7. Data Base Management System (DBMS) - система управления базами данных
8. Automated rollback - автоматизированная отмена изменений
9. Prompt (n.) - приглашение, вопрос
10. Data security - защита данных от несанкционированного доступа
11. Data dictionary - словарь базы данных
12. Entity - сущность, объект
13. Attribute - атрибут, свойство
14. Hierarchical data model - иерархическая модель данных
15. Object-oriented data model - объектно-ориентированная модель данных
16. Node - узел сети передачи данных или сети ЭВМ
17. Network data model - сетевая модель данных
18. Application Programming Interface (API) - прикладной программный интерфейс
19. Query - запрос, вопрос
20. Concurrency - параллелизм, взаимосовмещаемость
21. Data logging - протоколирование, регистрация данных
22. Atomicity - элементарность, атомарность
23. Structured Query Language (SQL) - язык структурированных запросов
24. Persistent - устойчивый, стойкий

What is a Database?

A **database** is a structured set of persistent data. A phonebook is a database. However, within the world of IT, a database is usually associated with software. A simple database might be a simple file containing many records, each of which contains the same set of fields where each field is a certain data type and length. In short, a database is an organized store of data where the data is accessible by named data elements.
A DBMS is a software package designed to create, store, and manage databases. The DBMS software enables end users or application programmers to share data. It provides a systematic method of creating, updating, retrieving, and storing information in a database. DBMS products are usually responsible for data integrity, data access control, automated rollback, restart and recovery.
You might think of a database as a file folder and a DBMS as the file cabinet holding the labeled folders. You implement and access database instances using the capabilities of the DBMS.

What is a Good Database Design?

Certain principles guide the database design process. The first principle is that duplicate information (also called redundant data) is bad, because it wastes space and increases the likelihood of errors and inconsistencies. The second principle is that the correctness and completeness of information is important. If your database contains incorrect information, any reports that pull information from the database will also contain incorrect information. As a result, any decisions you make that are based on those reports will then be misinformed. A good database design is, therefore, one that:

- Divides your information into subject-based tables to reduce redundant data.
- Provides access with the information it requires to join the information in the tables together as needed.
- Helps support and ensure the accuracy and integrity of your information.
- Accommodates your data processing and reporting needs.

The Design Process

The design process consists of the following steps:

- **Determine the purpose of your database**

It is a good idea to write down the purpose of the database on paper — its purpose, how you expect to use it, and who will use it. The idea is to have a well developed mission statement that can be referred to throughout the design process. Having such a statement helps you focus on your goals when you make decisions.

- **Find and organize the information required**

To find and organize the information required, start with your existing information. Each of the information items represents a potential column in a table. As you prepare the list, don’t worry about getting it perfect at first. Instead, list each item that comes to mind. If someone else will be using the database, ask for their ideas, too. You can fine-tune the list later.

Next, consider the types of reports or mailings you might want to produce from the database. Design the report in your mind, and imagine what it would look like. What information would you place on the report? List each item.

The key point to remember is that you should break each piece of information into its smallest useful parts.

- **Divide the information into tables**

To divide the information into tables, choose the major entities, or subjects. When you design your database, always try to record each fact just once. If you find yourself repeating the same information in more than one place, put that information in a separate table. Once you have chosen the subject that is represented by a table, columns in that table should store facts only about the subject.

- **Turn information items into columns**

To determine the columns in a table, decide what information you need to track about the subject recorded in the table. Once you have determined the initial set of columns for each table, you can further refine the columns.

- **Specify primary keys**

Each table should include a column or set of columns that uniquely identifies each row stored in the table. This unique identification number, such as an employee ID number or a serial number. In database terminology, this information is called the **primary key** of the table. Access uses primary key fields to quickly associate data from multiple tables and bring the data together for you. A primary key must always have a value. If a column's value can become unassigned or unknown (a missing value) at some point, it can't be used as a component in a primary key.

You should always choose a primary key whose value will not change. In a database that uses more than one table, a table’s primary key can be used as a reference in other tables. If the primary key changes, the change must also be applied everywhere the key is referenced. Using a primary key that will not change reduces the chance that the primary key might become out of sync with other tables that refer it

- **Set up the table relationships**

Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary. To represent **a one-to-many relationship** in your database design, take the primary key on the "one" side of the relationship and add it as an additional column or columns to the table on the "many" side of the relationship.

For each record in one table, there can be many records in another table. This type of relationship is called **a many-to-many relationship.** Note that to detect many-to-many relationships between your tables, it is important that you consider both sides of the relationship.

Another type of relationship is **a one-to-one relationship.** For each record in the table, there exists a single matching record in the supplemental table.

- **Refine your design**

Analyze your design for errors. Create the tables and add a few records of sample data. See if you can get the results you want from your tables. Make adjustments to the design, as needed.

- **Apply the normalization rules**

Apply the data normalization rules to see if your tables are structured correctly. Make adjustments to the tables, as needed.

Why Use a DBMS?

The main advantage of using a DBMS is to impose a logical, structured organization on the data. A DBMS delivers economy of scale for processing large amounts of data because it is optimized for such operations.
A DBMS can be distinguished by the model of data upon which it is based. **A data model** is a collection of concepts used to describe data. A data model has two fundamental components: its structure, which is the way data is stored, and its operations, which is the way that data can be manipulated. The major DBMS products utilize four different data models:
1. Hierarchical
2. Network
3. Relational
4. Object - oriented.

The hierarchical data model arranges data into structural trees that store data at lower levels subordinate to data stored at higher levels. Because a diagram of this model resembles an upside-down tree, it is often called a tree model. A hierarchical data structure, like a tree, consists of nodes connected by branches.
The network model is structured as a collection of record types and the relationships between these record types. All relationships are explicitly specified and stored as part of the structure of the DBMS.
The logical relationship between levels determines whether a network is classified as **simple** or **complex.** If the relationship is one-to-many, the network is simple. If it is many-to-many, the network is complex.
The relational data model consists of a collection of tables (more properly, relations) wherein the columns define the relationship between tables. The relational model is based on the mathematics of set theory. Contrary to popular belief, the relational model is not named after ‘relationships’, but after the relations of set theory. **A relation** is a set with no duplicate

values. Data can be manipulated in many ways, but the most common way is through SQL.
The object-oriented data model consists of a collection of entities, or objects, where each object includes the actions that can take place on that object. In other words, an object encapsulates data and process. With 00 systems, data is typically manipulated using an 00 programming language.
Each of these four data models is referred to as a data model for the sake of simplicity. In reality, only the relational and network models have any true, formal data model specification. Different models of data lead to different logical and structural data organizations. The relational model is the most popular data model because it is the most abstract and easiest to apply to data, while providing powerful manipulation and access capabilities.

Advantages of Using a DBMS

Additionally, a DBMS provides a central store of data that can be accessed by multiple users, from multiple locations. Data can be shared among multiple applications, rather than having to be propagated and stored in new files for every new application. Central storage and management of data within the DBMS provides

- Data abstraction and independence
- Data security
- A locking mechanism for concurrent access
- An efficient handler to balance the needs of multiple applications using the same data
- The ability to swiftly recover from crashes and errors
- Robust data integrity capabilities
- Simple access using a standard API
- Uniform administration procedures for data

Levels of Data Abstraction

A DBMS can provide many views of a single database schema. A view defines what data the user sees and how that user sees the data. The DBMS provides a level of abstraction between the conceptual schema that defines the logical structure of the database and the physical schema that describes the files, indices, and other physical mechanisms used by the database. Users function on the conceptual level - by querying columns within row of tables, for example - instead of having to navigate through the many different types of physical structures that store the data.
A DBMS makes it much easier to modify applications when business requirements change. New categories of data can be added to the database without disruption to the existing system.

Data Independence

A DBMS provides a layer of independence between the data and the applications that use the data. In other words, applications are insulated from how data is structured and stored. The DBMS provides two types of data independence:
- Logical data independence - protection from changes to the logical structure of data
- Physical data independence - protection from changes to the physical structure of data

As long as the program uses the API (application programming interface) to the database as provided by the DBMS, developers can avoid changing programs because of database changes.

Data Security

Data security prevents unauthorized users from viewing or updating the database. The DBMS uses IDs and passwords to control which users are allowed access to which portions of the database. For example, consider an employee database containing all data about individual employees. Using the DBMS security mechanisms, payroll personnel can be authorized to view payroll data, whereas managers could be permitted to view only data related to project history.

Concurrency Control

A DBMS can serve data to multiple, concurrently executing user programs. This requires a locking mechanism to deliver concurrency control because the actions of different programs running at the same time could conceivably cause data inconsistency. For example, multiple bank ATM users might be able to withdraw \$100 each from a checking account containing only \$150. A DBMS ensures that such problems are avoided because the locking mechanism isolates transactions competing for the same exact data.

Database Logging

The DBMS uses database logging to record ‘before’ and ‘after’ images of database objects as they are modified. It Is important to note that the database log captures information about every data modification. The information on the database logs can be used to undo and redo transactions. Database logging is handled transparently by the DBMS - that is, it is done automatically.

Ensuring Atomicity and Durability

A DBMS can be used to assure the all-or-nothing quality of transactions. This is referred to as **atomicity;** and it means that data integrity is maintained even if the system crashes in the middle of a transaction. Furthermore, a DBMS provides recoverability. After a system failure, data can be recovered to a state that existed either immediately before the crash or at some other requisite point in time.

Data Integrity

The DBMS provides mechanisms for defining rules that govern the type of data that can be stored in specific fields or columns. Only data that conforms to the business rules will ever be stored in the database. Furthermore, the DBMS can be set up to manage relationships between different types of data and to ensure that changes to related elements are accurately implemented.

Data Access

A DBMS provides a standard query language to enable users to interactively interrogate the database and analyze its data. For relational databases, this standard API is SQL, or Structure Query Language. However, SQL is not a requirement for a DBMS to be relational. Furthermore, many DBMS products ship with analytical tools and report writers to further simplify data access.

Before an installed DBMS can be used effectively, standards and procedures must be developed for database usage. A recent study conducted by Hackett Benchmarking & Research showed that companies with high levels of standardization reduce the cost of supporting end users by an average of 38% over companies with low level of standardization.
The Database Administrator (DBA) should develop database standards and procedures as a component of corporate-wide IT standards and procedures. They should be stored together in a central location as a printed document, in an online format, or as both.

Ex.l. Answer the following questions:

1. What is the difference between a database and a database management system?
2. What principles guide the database design process?
3. What are the main steps of the database design process?
4. What is the primary key of the table and what are its benefits ?
5. How can data in different tables relate?
6. What makes the relational data model the most popular?
7. Name and describe the advantages of using a DBMS.
8. What types of data models do you know?
9. How can atomicity prevent system crash?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

База данных - это упорядоченная совокупность связанных друг с другом данных. Система управления базами данных (СУБД) - это программное обеспечение, предназначенное для организации данных таким образом, чтобы облегчить и ускорить доступ к данным. С помощью СУБД пользователь может создавать, модифицировать, сохранять и получать данные различными способами.
С базами данных обычно связывают несколько преимуществ. **Уменьшенная избыточность данных.** Данные, если они хранятся в отдельных файлах, в сравнении с базами данных, имеют тенденцию дублировать часть информации снова и снова. В базах данных какие-либо сведения обычно появляются один раз.
Интегрированные данные. В отличие от информации, хранимой в отдельных и независимых файлах, сведения в базах данных рассматриваются как связанные друг с другом,

так как любая порция данных может быть использована при выполнении запроса или составлении отчета.

Целостность. Проблемы обеспечения целостности данных возрастают с ростом сложности организации данных. По этой же причине сокращается избыточность данных.

Способ, которым базы данных организуют информацию, зависит от типа или модели данных. Существуют три модели данных - иерархическая, сетевая и реляционная. Базы данных персональных компьютеров обычно реляционные.

Реляционная база данных организует информацию в форме таблиц, состоящих из связанных строк и столбцов. В реляционной системе данные одного файла могут быть связаны с данными другого файла, позволяя соединять вместе данные из нескольких файлов.

В реляционной базе данных таблицы называются отношением. Отношение также называется файлом. Каждый сегмент таблицы содержит часть данных, называемых минимальная единица информации. Конкретное значение минимальной единицы в поле может изменяться, но каждое поле хранит один и тот же тип данных. Все данные в отдельной строке называются записью. Каждая запись имеет фиксированное число полей, но может существовать различное число записей в данном отношении.

Ex.4. Topics for discussion.

1. The difference between a database and a database management system.
2. The advantages of using a DBMS.
3. The types of data models and their advantages.
4. The steps of the database design process.

Systems Analysis and Design

Key vocabulary:

1. Feedback - обратная связь
2. Information system - информационная система
3. Systems development - разработка системы
4. Systems analysis - системный анализ
5. Systems design - проектирование системы
6. Implementation - внедрение системы
7. Structured systems analysis - структурный системный анализ
8. Data dictionary - словарь данных
9. Decision tree - дерево решений
10. Goal oriented system - целевая система
11. System approach - системный подход
12. Evaluation - оценка
13. Maintenance - техническое обслуживание, поддержка
14. Artificial data - искусственные данные
15. Conversion - преобразование, конверсия
16. Dual system method - метод дублирования
17. Inventory method - метод полной замены системы
18. Structured design - структурированное проектирование
19. Training - обучение (людей), отработка (системы)
20. Session - сеанс
21. Parallel systems method - метод параллельных систем
22. Design documentation - документация по проектированию
23. Training documentation - учебная документация
24. User reference documentation - документация для пользователя

What is a System?

What do a living cell, an automobile, NASA's Mission Control Center, and a university have in common? The answer is that they are all systems.

These systems are, of course, very different. A cell is made up of cytoplasm, membranes, and DNA, while an automobile contains metal, plastic, lubricants, and glass. But both automobiles and cells contain a variety of parts that work together. The university and Mission Control also contain a variety of parts - in these cases, people, equipment, and buildings. But these systems also have specific goals: a university educates people, Mission Control directs space flights. In other words, these **systems** are more than a set of parts or elements that interact. They exist to achieve a specific **goal.**

The goal of a system usually involves transforming various kinds of input into a desired output. In a university the inputs are teaching and related services, the output is educated people and research. In a computerized business system inputs are raw data, and the output is likely to consist of information in a form needed by management for planning and decision-making.

Once the system has been put into operation, it should do the job management wants it to do. If it doesn't, people who use it will, probably, complain and the system will be either modified or scrapped. The evaluations of users - 'it works, keep it' or 'it doesn't work, throw it out' - are a primitive form of feedback. **Feedback** is information about how well a system works, it is used to maintain or improve a system.

A simple example of feedback is a thermostat that controls a heating system. The inputs to the system are the thermostat setting and fuel for the heating unit. The output is the actual heating required to reach that setting. The thermostat continually

Records the temperature of the room as the output modifies it, it reports this temperature to the system as feedback. When the temperature matches the thermostat setting, the system modifies itself by ceasing its output, that is, turning off the heat.

The computer itself is a system and an organization within which a system operates is also a system. Systems that are part of a larger system are called **subsystems.**

A modern corporation, which meets our definition of a system, contains such subsystems as finance, personnel, production, research and development, and advertising.
What does it matter that everything is a subsystem of something else? In order to design an improved system you have to pin down as precisely as possible what it is that you are dealing with. You must first analyze the system and its components. Taking *a systems approach* will tell you two things: first, you will see how a subsystem fits into a larger system of which it is a part. Second, you will see how the tasks of a system break down into sub-systems. Only then will you be able to study the interactions among them to find out which interactions can be improved.
Information system is a collection of software applications that store and manipulate information. Like any computer system, an information system has a lifetime - it is born, lives for some time, and dies. Many systems follow the same basic pattern, or *life cycle.*
The stages of the information system life cycle are: analysis, design, and implementation.

Systems Analysis

Systems analysis is the first stage of **systems development.** It consists of defining a problem to be solved or an opportunity to be taken and identifying the objectives for a new system.
Systems analysis proceeds in a series of steps:
1. Define the problem or opportunity
2. Collect data
3. Analyze existing methods and procedures
4. Identify objectives and opportunities
5. Present a proposal to management

Define the Problem or Opportunity. Once a problem or opportunity has been pointed out to the systems analyst, the first task is to draw the boundaries of the system involved. The result of the first step in systems analysis is usually a preliminary report outlining the score of the problem and making a recommendation, which may be that a more detailed study is needed, or that it is (or isn't) feasible to proceed with the next stage of analysis.

Collect data. Just as a detective collects clues, the systems analyst must collect samples of records, documents, or usage patterns. The analyst doesn't collect data about some hypothetical future systems; the need is for data about the present system, with all its problems and shortcomings. This data is used to document all the relevant activities of the system under study.
Systems analysts usually collect data by observation, interviews, and questionnaires.

Analyze Existing Methods and Procedures. After defining the problem and collecting data, the systems analyst must figure out exactly how the current system works in order to see what, if anything, is going wrong.

Identify Goals and Opportunities. At this point the analyst must outline, in measurable terms, what the system should be able to do. It is important to state the goal clearly and strongly. In stating goals, it is important to think not only in terms of improving the present system, but also in terms of related benefits or opportunities.

Present a Proposal to Management. The last step in systems analysis is to make a formal presentation of findings and recommendations to management. The report should contain a menu of alternatives - a list of possible solutions to the problem. The alternatives all solve the problem, or take advantage of the opportunity, that was stated in the problem definition, but they differ in significant ways. Users and management select what they think is the best alternative on the basis of the trade-off between costs and benefits.

Systems Design

In systems design, the first step is to consider a logical model for the proposed system and the set of objectives the proposed system should meet. Then specifications are produced for a physical system to meet those objectives. In other words, in this stage of systems development the processes are designed that are needed to convert the input of the system into output that achieves the system's goals.

Systems design can begin only after systems analysis has been completed, since it is the analysis that creates the model and set of objectives that users, analysts, and management agree on. However, systems design must come before implementation. Programmers sometimes think they can work out the design while coding their programs, but like authors who think they can write a book without making an outline in advance, they generally turn out results that are less than ideal.

An objective of a good design is to produce a flexible system, one that can be modified easily. A system that is easy to modify need not be thrown out in its entirety when something goes wrong. A flexible system can also be adapted and built on more readily than an inflexible one. Another important part of structured systems design is to design a system at *the logical* level (what the system must do) before committing oneself to a *physical* solution. A logical design is easy to understand, since it uses graphics, omits technical details, and aims at brevity.

A key task in logical design is to create a changeable system. The reason for this is that maintenance, or making changes or enhancements to existing system, still accounts for about 70 to 80 % of the time spent in data-processing shops. Thus more time is spent modifying existing systems than is spent in creating new ones.

Systems Implementation

Implementation is the final phase of systems development. In this phase the system is actually created, tested, and documented. The outcome of this stage is the successful placement of the sys-tem within the organization's day-to-day operations.
There are five distinct steps in system implementation: programming, **testing, training, conversion, and documentation.** Programming is itself a complex step and a major topic.

Testing. A computer system consists of one or more interacting programs. We test a system by running each component program in succession. We want to be sure that each program's input and output fit into the total system in the way the system's designers intended. This is when the 'bugs' - unintended errors or problems
- should be found and removed.
Artificial data is usually created for running the first tests. It
should include deliberate mistakes that help us check the system.
Once the system performs flawlessly on artificial data, we switch
to 'live' data taken from the organization.
A system is generally tested in a hierarchical fashion, starting at the bottom and working up. First, each program module is tested; next a series of modules is tested; then each individual program with all its modules; finally, the entire system, consisting of a series of programs, is tested.

Training. A system might be flawlessly designed and tuned to perfection, like a fine car, but in the hands of a user who can't get it into reverse gear or panics at the sight of a traffic lights, it is useless. All the users who will be affected by the new system should be trained in its use. Training includes an overview of how the system functions, how it will affect their jobs, and how it will relate to current manual procedures.
Training should be planned in advance, and the plans should be approved by the management. Common sense would dictate that

training sessions not be scheduled during high-pressure periods, such as when the system itself is being tested or during the busiest time of the day.

Conversion. Once a system has been tested successfully and users have been trained in its use, the old system can be converted to the new one. System conversion can be approached in several ways: the parallel-systems method, the dual system method, and the inventory method.
In *the parallel-systems method*, the new system is set to work alongside the old one. Data is input to both simultaneously, until the new system has demonstrated that it functions effectively.
In *the dual-system method* the old system is gradually phased out while the new one is being phased in.
The inventory method is a complete, 'cold turkey' (abrupt) changeover from the old system to the new one.

Documentation. Anything that is written about how a system is designed or functions is documentation. The documentation should be adequate, clear and intelligible to its audience, and current.

Evaluation. Once a system is installed and running and all the 'bugs' have been removed, it should be evaluated to see how well it does its job. This might be done, for instance, within a year after implementation and once every three years thereafter.

Ex.l. Answer the following questions:

1. What is a system?
2. What is the interrelation between a system and a subsystem?
3. What do you think 'feedback' means?
4. Name the stages of systems development.
5. What steps in systems analysis do you know?
6. How do systems analysts collect data?
7. What is the objective of a good systems design?

8. Implementation is the final phase of systems development, isn't it?
9. Name the steps in systems implementation.
10. Why is it necessary to evaluate a system?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

1. Обратная связь - это информация о том, как работает система.
2. Информационная система - это метод, применяемый для сбора, доставки и использования необходимой информации.
3. Система разработки программ - это циклический процесс, который включает в себя анализ, проектирование системы и реализацию.
4. Системный анализ - первая стадия разработки программы. Она состоит из серии шагов: определение задачи, сбор данных, анализ существующих методов, определение целей.
5. Структурный системный анализ - это пошаговая модель какой-либо системы. Он подразумевает тесное взаимодействие между аналитиками и пользователями системы.
6. Цель проектирования - создание гибкой системы, которую можно легко изменить и адаптировать.
7. Реализация - конечная ступень разработки системы. В этой фазе система фактически создана, проверена и имеет документацию.

Ex.4. Topics for discussion.

1. The characteristics of a system and a subsystem.
2. The stages of the systems development process, including systems analysis, design, and implementation.
3. The characteristics of structured design.
4. The steps in the implementation process.

Management Information Systems

Key vocabulary:

1. Management Information System - информационно-управляющая система
2. System Development - разработка системы
3. A rigorous method - строгий, точный метод
4. A blueprint - наметка, план, проект
5. Centralized data processing - централизованная обработка данных
6. Decentralized data processing - децентрализованная обработка данных
7. Distributed data processing - распределенная обработка данных
8. Consistency of data - согласованность, постоянство, непротиворечивость данных
9. A backlog - незавершенная, невыполненная работа
10. A backup file - резервный, дублирующий файл
11. Prime advantage - важнейшее, основное преимущество
12. In-depth knowledge - основательность, глубина знаний

Systems Development Methodology

The need for more precise systems development methods grew out of the problems that arouse when less rigorous methods were used. Errors made in the analysis and the design phases of a project often were not caught until the implementation phase, when corrections had to be made at a cost many times grater than if such changes had been made earlier. A lack of communication with end users at the analysis and the design phases often led to the development of systems that didn't meet user's needs. As a result, major modifications and enhancement had to be designed

after the system was implemented, causing substantial time and cost overruns.
These problems influenced the growth of a more precise systems development process.
During detailed systems analysis and design, the ‘blueprints’ for the new system are drawn. These ‘blueprints’ include procedures, program specifications, and design of the database, as well as a plan for the next phase, implementation.
If the implementation plan is approved, then the actual development of the system, its programs, databases and procedures can begin. Program specifications and programs are designed according to a predetermined set of standards. The systems development methodology also includes maintenance - the effort to keep the information system up to date with new technologies, and to serve changing corporate requirements.
Finally, the system is subjected to periodic review and auditing to make sure it continues performing as desired. Feedback from this type of periodic review provides management with the opportunity to improve, modify, and enhance the system if necessary.
Data processing operations in a corporate or other institutional setting can be organized in one of several ways: **centralized, decentralized, and distributed.**

Centralized Data Processing

The hardware for a centralized data processing installation generally consists of a large processing unit supporting a variety of peripheral devices. The system can process more that one application program, often simultaneously, and can handle various kinds of input and output.
A centralized computer system may support remote terminals, scattered about a business, school, or a city. Although this gives the appearance of decentralized data processing - and is, in reality, decentralized in terms of *access* to computer resources -

control of those resources, their priorities, the charges for them, and so on, are still determined by the central facility that does the actual computing.
Centralized data processing organizations maintain a permanent staff of highly trained specialists. These individuals select the hardware, manage the operating-system environment, set up and manage databases, arrange telecommunication facilities, run the computers and its peripheral devices, perform systems analysis and design, and do the actual programming.
Advantages of Centralized Data Processing. The first advantage is ‘the economy of scale’ - that is, centralized facilities aim to satisfy the diverse computing needs of an entire organization with a single facility, thereby saving the costs of multiple facilities. Second, centralized data processing centers can keep all the files needed by an organization in one place, which can increase the consistency and standardization of the data.
Closely tied to this advantage is the great security and tight control over the access to the data that housing it in a central facility permits. In a large data processing center, a database administrator is responsible for keeping backup copies of data files, so that data is not lost, and also for distributing passwords and providing other security precautions, so that sensitive data is not accessible to persons not permitted to use it.
The advantage of any sort of ‘center’ is that users of many types, including those in remote divisions of a firm, can have access to important data and the use of computing power without having to set up such facilities themselves. Moreover, ‘centers’ attract computer personnel more readily because the applications that are designed and maintained are more sophisticated and because they offer multiple career paths, in-house training, and upword mobility.
Another point in favor of centralization is its ability to control the overall cost of data processing on an organization-wide basis.
Disadvantages of Centralized Data Processing. Naturally, centralized data processing has its drawbacks. First, forcing all

systems development to flow through a central facility can lead to backlogs when the existing staff is overloaded. It is not uncommon in large corporations to find projects that management approved for development as long as two years ago still waiting for the centralized facility to get started on development.
Closely related to this first advantage is the next - small projects may not get done at all.
The third main drawback is the lack of user control over the development of computer-based information systems and also over their day-to-day operations. Computers are meant to serve their users, but in trying to serve a great number of users - as in the center - responsiveness to individual application areas may be lessened.

Decentralized Data Processing

For a decentralized data processing installation, the hardware generally consists of a single minicomputer or microcomputer and its associated peripheral devices. The system usually processes only programs for a specific application area, although the computers themselves are capable of handling a wide variety of applications.
A decentralized installation is self-contained, with its own operating system and DBMS. The data files kept by the decentralized installation will be used only by that computer, and may not be compatible with the organization's central mainframe computer or with other decentralized installations in the organization. The software used by a decentralized installation may be simple, not offering many options outside the application's specific needs.
Decentralized computers usually need at least one person to handle the day-to-day operations of the computer. This person might be a computer professional, or a secretary or a clerk. If packaged software is used - which is purchased ready to run from outside vendors and can be tailored to individual requirements by outside firms as well - then a programmer is not necessary.

Advantages of Decentralized Data Processing. Local autonomy, of course, is the prime advantage of decentralized data processing. With a machine on hand, and the availability of appropriate software assured, users are freed from dependence on a centralized facility that may not be responsive to their particular needs. Moreover, once the computer has effectively taken over the processing of the particular application for which it was intended, it probably has the capacity and ability to take on new tasks the department or user can devise. This responsiveness to new and changing user application needs is frequently missed in centralized processing situations.

Disadvantages of Decentralized Data Processing. One disadvantage is loss of control of the organization's data processing operations by centralized management. Users may see this as an advantage, but the corporation can take a different view. A proliferation of decentralized facilities is a step in the wrong direction when it comes to providing access to data for various corporate users. With incompatible systems scattered about an organization, the machines cannot share data or be hooked together into a network.

From a computer professional's standpoint, decentralized installations may be inefficiently run. The application systems may be designed without adequate standards, data security, backup files, or documentation.

Decentralized computers may also lead to an inefficient use of personnel. The purchase of identical software packages, and even the development of identical programs in different departments, is one more type of inefficiency.

Distributed Data Processing

In distributed data processing, a communications network connects centralized and decentralized computers, which are compatible with each other. Local mini- or microprocessors, with local data files and local processing ability, can interact with a

central computer system to obtain access to central data files or to transmit information to the central system.
Responsibilities for data processing systems development and operations in distributed data processing are shared between centralized and decentralized personnel. Centralized personnel set the standards for the communications network and establish interface requirements to the central system, including methods of accessing centralized databases. They may define standards for project selection, project control, and management.
Data processing personnel in decentralized sites have responsibility for determining local application development priorities, for managing local data processing operations, and for maintaining local hardware and software.
Advantages of Distributed Data Processing. One of the benefits of distributed data processing is the ability to offload work from the central computer system and in this way reduce processing costs. If data can be validated locally, or if transactions can be recorded and files updated locally, the costs of communications to the central system, as well as the costs of processing, are substantially reduced.
Distributed data processing not only decentralizes some processing but also decentralizes some of the responsibility for data processing activities, including systems development and operations. Local systems analysts, who are familiar with local problems and priorities, are more likely to be responsive to user needs.
Most importantly, in distributed data processing local management can better understand and directly control data processing costs.
Disadvantages of Distributed Data Processing. Users in remote sites may reinvent the wheel by approving systems projects that already exist and in this way actually add to overall corporate data processing expense. Central data processing personnel may feel a lack of control over locally approved projects and may question their justification. Systems projects that are completed may not

conform to central systems and programming and documentation standards, making more locally developed applications difficult to maintain. In addition, local procedures for data security and contingency planning may be inadequate compared with central plans.

How to Organize. One factor that affects the decision of how to organize data processing is how the organization is structured. A conglomerate of decentralized divisions will likely place considerable computing power in the hands of these divisions. However, the central corporation will maintain some control over operations, as a way of monitoring EDP expenses.

Control of computing operations is a form of power, and people have seldom shown a reluctance to acquire it or to prevent others from doing so. In the case of centralization versus decentralization, this may mean that large divisions will prefer a decentralized approach because this enhances their own power, smaller divisions may favor centralization because they can benefit from the expertise of the central facility.

Ex.l. Answer the following questions:

1. What problems required more precise systems development methods?
2. What are the steps of a systems development process?
3. What are the advantages and disadvantages of centralized data processing?
4. What are the advantages and disadvantages of decentralized data processing?
5. How does distributed data processing help to reduce data processing costs?
6. What are the disadvantages of distributed data processing?
7. What affects the decision of how to organize data processing?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

Различаются следующие способы обработки данных: **централизованный, децентрализованный, распреде-ленный и интегрированный.**

При **централизованном способе** пользователь доставляет на ВЦ исходную информацию и получает результаты обработки в виде результативных документов. Особенностью такого способа обработки являются сложность и трудоемкость налаживания быстрой, бесперебойной связи, большая загруженность ВЦ информацией (т.к. велик ее объем), регламентация сроков выполнения операций, организация безопасности системы от возможного несанкционированного доступа.

Децентрализованная обработка. Этот способ связан с появлением ПЭВМ, дающих возможность автоматизировать конкретное рабочее место.

Распределенный способ обработки данных основан на распределении функций обработки между различными ЭВМ, включенными в сеть. Этот способ может быть реализован двумя путями: первый предполагает установку ЭВМ в каждом узле сети (или на каждом уровне системы), при этом обработка данных осуществляется одной или несколькими ЭВМ в зависимости от реальных возможностей системы и ее потребностей на текущий момент времени. Второй путь - размещение большого числа различных процессоров внутри одной системы. Такой путь применяется в системах обработки банковской и финансовой информации, там, где необходима сеть обработки данных (филиалы, отделения и т.д.). Преимущества распределенного способа: возможность обрабатывать в заданные сроки любой объем данных; высокая степень надежности, т. к. при отказе одного технического средства есть возможность моментальной замены его на другой; сокращение времени и затрат на передачу данных; повышение гибкости систем, упрощение разработки и эксплуатации программного обеспечения и т.д. Распределенный способ основывается на комплексе специализированных процессоров, т.е. каждая ЭВМ предназначена для решения определенных задач, или задач своего уровня.

Интегрированный способ обработки информации. Он предусматривает создание информационной модели управляемого объекта, т.е. создание распределенной базы данных. Такой способ обеспечивает максимальное удобство для пользователя. С одной стороны, базы данных предусматривают коллективное пользование и централизованное управление. С другой стороны, объем информации, разнообразие решаемых задач требуют распределения базы данных. Технология интегрированной обработки информации позволяет улучшить качество, достоверность и скорость обработки, т.к. обработка производится на основе единого информационного массива, однократно введенного в ЭВМ. Особенностью этого способа является отделение технологически и по времени процедуры обработки от процедур сбора, подготовки и ввода данных.

Ex.4. Topics for discussion.

1. The organization of the systems development process.
2. Centralization and decentralization of some functions by certain organizations.
3. Splitting data processing functions at an installation into system development, operations and technical support.
4. Distributed data processing - its pluses and minuses.

Decision Support Systems and Artificial Intelligence

Key vocabulary :

1. Artificial intelligence (AI) - искусственный интеллект
2. Exception report - сообщение об ошибках, исключительных ситуациях
3. Enquiry system - справочно-информационная система
4. Decision-support system - система поддержки принятия решений
5. Spreadsheet - динамическая электронная таблица
6. Ad-hoc system - произвольная система, система на данный случай
7. Mockup - модель, макет
8. Expert system - экспертная система
9. ‘Rule of thumb’ - железное правило, эмпирическое правило
10. Knowledge base - база знаний
11. Inference engine - механизм вывода
12. Knowledge acquisition - пополнение, приобретение знаний
13. Forward chaining - вывод ‘от фактов к цели’
14. Backward chaining - вывод ‘от цели к фактам’
15. Robotics - робототехника
16. Tactile sensing - сенсорное управление

A small business needs information systems to support day-to-day operations. These information systems streamline the paper-work involved in processing day-to-day transactions, such as orders, bills, and payments.

Many managers confront problems that are unstructured and that take into. account many complex variables. These complex problems require expert knowledge, which in the past was only avail-able from experts themselves. Tools available today make it possible to capture expert knowledge and to program computers with this knowledge to support effective decision-making. Expert

systems incorporate ‘expert’ knowledge that can be used to solve problems in medicine, engineering, and in business. The field of expert systems is a part of a larger field of artificial intelligence, which is concerned with the creation of computer programs to do things in ways more like human patterns of thinking than ‘traditional’ computer programs.
One way of providing managers with information is to develop **exception reports,** which show significant deviations from planned activity.
The only problem with exception reports is that the exception conditions are fixed. The exception-reporting systems were not supporting important decision-making needs, but there was a tremendous demand for enquiry systems. **Inquiry systems** provide a database with flexible inquiry capability, making it possible for managers to ask a database questions. Inquiries can change on a day-to-day basis, and the inquiry systems managers are effective in supporting their information needs.

The Characteristics of Decision Support Systems

A **decision support system** can be defined as a computer-based system that helps decision-makers use data and models to solve unstructured problems. Decision support systems are designed to support inquiry and analysis needs. A decision support system requires a database that serves as a repository of data for managers to use in making queries and in generating reports.
Another characteristic that distinguishes decision support systems from information systems is their basis for justification.
A good decision support system has to take into account the characteristics of decision-making and decision-makers. A decision support system must support unstructured decisions, all phases of the decision-making process, and communications between decision-makers. In addition, an effective decision support system must provide memory aids and training facilities that make its tools easy to learn how to use.

Decision-making can be divided into three phases: intelligence, design, and choice. During the first phase, **intelligence,** a situation requiring a decision is visualized. During the second phase, **design,** the problem is defined and the alternative solutions are evaluated. The third phase, **choice,** results in the selection of the proper course of action.

The tools used to support each of these phases vary because the operations that need to be conducted vary. For example, a data-base package can be used to generate data needed at the intelligence phase. A spreadsheet package may be more useful in examining a series of alternatives at the design phase. A statistical analysis or a graphics package could be used to demonstrate the results of various alternatives during the choice phase.

Decision-makers need memory aids, or information about the results of operations that were conducted at previous times. Workspaces for storing the intermediate results of operations are very useful.

Training aids are important, particularly in the early months of system used. Decision support tools should provide menus, help screens, and prompts, which ease the learning process and help users develop language skills.

Decision support systems are generally developed by user managers who decide to use a database, spreadsheet or other program to support a business need. Each decision support system sup-ports important departmental objectives, it constantly evolves into an information system that can be used for analytic purposes as well as for query and reporting. The final characteristic of a decision support system is that it is often used and developed by business professionals who have very little formal training. The major motivation for developing these decision support systems is a business need or a competitive strategy.

The Development of Decision Support Systems

These systems are in constant evolution. The strategy used to develop a decision support system involves interaction between a

builder and a user. At the start a user defines a problem area and begins to think about the information needed. The builder works with the user to come up with some ‘mockup’ versions of reports from the system. The user tries these reports out, and they are continually modified until the correct information has been defined. This approach encourages the development of short-lived, ad-hoc systems that can be constantly refined. Continuing user feedback brings about new versions of the system.
The major steps in the DSS development process are contrasted with the major steps in the traditional development process. The first phase, **planning,** involves the user in diagnosing a problem that can be solved through the development of a decision support system. After planning the user considers what development approaches and tools are appropriate for the project. This phase is called **application research.** During **analysis** the best approach is selected, and during **design** the detailed specifications for the system are established. **System construction** is the technical implementation of the design. The entire effort may be considered prototyping: the system that is constructed is treated as a model, or prototype, and it can be constantly reevaluated and modified.

The Characteristics of Expert Systems

Expert systems are systems that simulate the knowledge of experts on specialized, professional tasks. Expert systems are taught ‘the rules of thumb’, or heuristics, that experts use to solve problems. These rules can be continually updated as the situation changes.
An expert system solves difficult problems in much the same way an expert consultant would, by asking the user for information and relating this information to general rules. If it needs additional information, it asks more questions until it arrives at conclusions or makes recommendations. An expert system is composed of **a knowledge base, an inference engine, an explanation**

subsystem, a knowledge acquisition subsystem, and a human interface component.

The knowledge base. The expert system has a knowledge base that stores the factual knowledge and the rules of thumb it needs to make decision. One of the ways of representing knowledge is to use rules. In a rule-based system the knowledge base is a collection of facts and inference rules. The facts deal with the system's specialized area of expertise and are applied to specific problems as needed.

The inference engine is the 'CPU' of the system that that uses the knowledge base to draw conclusions for each situation. The inference engine conducts a dialogue between the user and the knowledge base. The most commonly used inference methods are data-driven, goal-driven, and mixed. In the data-driven method, which is also known as **forward chaining,** the user enters a series of facts and the program responds by asking questions. The goal-driven method uses rules that apply only to a particular goal. Otherwise known as **backward chaining,** this method proceeds backward through the rules to try to prove a goal. The mixed method combines the data- and goal-driven methods.

The explanation subsystem explains the line of reasoning that has been used in arriving at a point in the decision-making process or at its recommendations. Explanations the system provides can explain the strategies used to solve the problem at hand, can respond to specific questions of the user, and can criticize the solutions being proposed.

The knowledge acquisition subsystem makes it possible for new rules to be added to the knowledge base. Because of this knowledge acquisition component the expert system can be developed in an ongoing process.

The human interface. Expert systems are designed to be used by business, engineering, and other professionals - not by technical experts. As a result, these users need to be able to communicate with the system and to understand its feedback. Since most expert

systems are designed to deal with a narrow area of knowledge, the vocabulary they use should be familiar to their users.
Many of the earlier expert systems were designed in medical diagnosis. Using knowledge of a patient history, symptoms, laboratory test results they diagnosed the cause of the illness and selected appropriate therapy.
In the area of geology many expert systems have been developed to aid in the exploration and drilling of oil wells.
Many successful expert systems have been designed to diagnose faults in electronics systems, in telephone networks, in the design of digital circuits
There is also great potential to develop commercial applications of expert systems: there is an expert system for credit authorization, an expert financial planner.

The Development of Expert Systems

Knowledge is inexact, and no one has complete knowledge. Even though the expert's knowledge is often 10 times more complete than the knowledge of amateurs, his or her answers are still only about 70 to 80 percent certain. An expert system is no better than an expert and will never be able to offer knowledge that is completely certain.
Secondly, knowledge is incomplete. It is usually acquired incrementally, through trial and error. Expert systems are also developed in an incremental way. Requirements are not all defined in advance. Normally a prototype is developed and rules are added as new situations occur.
Expert knowledge usually applies to a very specific area. We know of expert auto mechanics, expert chefs, and expert tennis players. No one is an expert doctor, but a doctor may be an ex-pert in diagnosing infectious blood diseases, for example. No one is an expert lawyer, but a lawyer may have expertise in the area of maritime law. Problems with very few rules (fewer than 10) are not good candidates for expert systems development because they can be handled well by most humans. An expert system cannot

have common sense, so problems requiring common sense are not good candidates either.
The problems suitable for expert systems development include those requiring analysis and synthesis
An expert system can serve well as a consultant or specialist in the absence of a human consultant. The expert system can combine the knowledge of several different specialists and actually be superior to the expertise of a single consultant. Expert systems have greater consistency in dealing with problems because they don't forget relevant factors under the constraints of stress and time. They can also arrive at faster solutions and generate a greater number of alternatives. In addition, an expert system can be recorded and used for consultation and training purposes in multiple locations, simply by copying a disk file.

Expert Systems and Decision Support Systems

Expert systems and decision support systems both support problem-solving in an unstructured environment. Expert systems make it possible to expand the flexibility of the problem structure. They also offer the use of an inference engine, a knowledge base, and a knowledge acquisition system. A DSS provides the user manager with alternatives for consideration, but an expert system can help the user evaluate these alternatives or may even prompt the decision-maker with approaches or choices that have been employed in other situations. The expert system can serve as a consultant, providing 'rules of thumb' for situations that deal with incomplete, uncertain data. In the future, many managers may use expert system shells to augment the decision support systems they have already developed.

Robotics

The application of artificial intelligence that has been a major part of factory automation is robotics. Most robots handle specialized tasks, such as cutting, drilling, painting or welding. Most are

designed to do a single function, following the same pattern of motion day after day.
Although most robots are programmed to perform a simple, repetitive task, some robots perform a series of operations.
One problem in robotics is that robots cannot be programmed to move exactly according to the computer's instructions. One way of dealing with this problem is to endow robots with sensors, such as machine vision or tactile feedback. When a sighted robot misses its target, it can see its error and adjust its position accordingly. Vision systems are also used to enable robots to identify objects correctly so that they know what stored programs to use.
Tactile sensing is also an ability of robots.
In the past, robots were used primarily for jobs in which humans faced safety and health hazards. In general, robots are slower and more expensive than people. They are much better equipped to perform a single task day after day than they are to handle complete operations.
With the dramatic growth of technology in the artificial intelligence area, we might feel that it will not be very far into the future before fiction becomes fact, and expert systems are able to read a human's thought processes and to do the reasoning for him or her. However, before we begin to think that computers may become intelligent in and of themselves, we need to rethink their potential. Computer technology is only as powerful as we can design it to be. It is only as useful as our requirements dictate. User designers, knowledge engineers, and system builders in the future will have a growing role in business, industry, and the service professionals. Thus, there is a need to control the effective use of these technologies so that they can be used productively and successfully to solve business, scientific, and medical problems.

Ex.1. Answer the following questions:

1. What are the ways of providing the managers with information?

2. What distinguishes a decision support system from an information system?
3. Haw can a good decision support system be created?
4. What are the stages of decision-making?
5. What are the characteristics of a decision support system?
6. What are the steps in the DSS development process?
7. How can an expert system be defined?
8. Is there any difference between expert systems and decision support systems?
9. Can robots today be as smart as humans?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

Термин “искусственный интеллект” (ИИ) употребляется сегодня в двух основных смыслах. С одной стороны, ИИ понимается как нечто, присущее определенному классу компьютерных систем, работа которых может быть понята как существенно сходная с тем типом человеческой деятельности, который мы называем интеллектуальным. Степень этого сходства и правомерность квалификации компьютерной системы как интеллектуальной в полном смысле слова - предмет дискуссий, которые начались с появлением первых систем, названных интеллектуальными и ведутся по сей день.
С другой стороны, термином ИИ обозначается научно-техническое направление, в рамках которого исследуются, изобретаются и реализуются различные способы создания компьютерных систем.
Имеющиеся в настоящее время системы искусственного интеллекта (ИИ) способны выполнять функции, ранее считавшиеся исключительно прерогативой человека: переводить тексты с одного языка на другой, доказывать математические теоремы, диагностировать болезни,

распознавать месторождения полезных ископаемых, умело играть в шахматы и другие интеллектуальные игры.
Роботы, эти автоматы, наделенные ИИ, еще недавно бывшие лишь персонажами фантастических рассказов, в наши дни уже реально существуют, выполняя без участия человека - оператора целенаправленные действия, опасные или невозможные для человеческого организма. Отличие современных роботов от ранее известных автоматов состоит в том, что они обладают такими "интеллектуальными способностями", как способность обучаться, приспосабливаться к изменяющейся среде обитания, действовать целенаправленно, "осмысленно" имитируя поведение человека.
Еще один важный класс систем ИИ, получивший распространение в последнее время, - это т.н. "экспертные системы", т.е. системы, позволяющие на базе современных компьютеров выявить, накапливать и корректировать знания из различных предметных областей.

Ex.4. Topics for discussion.

1. The characteristics of decision support systems.
2. The development of decision support systems.
3. The characteristics of expert systems.
4. The difference between decision support and expert systems.

Some Hints on Comparing Brain and Computer Processing

Key vocabulary:

1. Initiation - включение, запуск, создание, инициация
2. Pipelining - конвейерный режим, конвейерная обработка
3. Hyperthreading - технология гиперпоточности
4. Bandwidth - пропускная способность, диапазон частот
5. Single sourced output - централизованный вывод
6. Interleaved - чередующийся, перемежающийся
7. Overlay - наложение, перекрытие, оверлей
8. Overwrite - наслоение, перезапись, наложение новых данных
9. Ample capacity - усиление, расширение объема (памяти)
10. Feedback loop - петля, контур обратной связи
11. Human-computer interface (HCI) - интерфейс ‘человек-машина’
12. Stand-alone - независимый, автономный
13. Information system/information technology (IS/IT) - информационная система / информационная технология
14. Cortex - кора головного мозга
15. Corpus callosum - мозолистое тело, пучок нервных волокон, соединяющий оба полушария головного мозга
16. Retinal - относящийся к сетчатке глаза
17. Supplant - вытеснять, выжимать, заменять

Over thirty years ago, TV shows from The Jetsons to Star Trek suggested that by the millennium’s end computers would read, talk, recognize, walk, converse, think and maybe even feel. Since people do these things easily, how hard could it be? Yet today we generally still don’t talk to our computers, cars or houses, and they still don’t talk to us. The Roomba, a successful household robot, is a functional flat round machine that neither talks to nor knows its owner. Its “smart” programming mainly tries not to get

"stuck", which it still frequently does, either by jamming itself under a bed or tangling itself in carpet tassels.
Computers do easily calculation tasks that people find hard, but they have difficulty in recognizing patterns. Other tasks easy for people but hard for computers include language recognition, problem solving, social interaction and spatial coordination. While people recognize familiar faces under most conditions, computers struggle to recognize known terrorist faces at airport check-ins because variations like lighting, facial angle or expression, or accessories like glasses or hat, upset the computer's fragile logic. Advanced computers struggle with skills most five-year-olds have already mastered, like talking, reading, conversing and running:

Computer vs. Human Information Processing

In comparing human and computer systems, the brain corresponds to a computer's central processing, not its printer or screen. The brain's trillion (1012) neurons operate by electricity, and like transistors are on/off devices that allow logic gates In the neurocomputational approach neural patterns encode, trans-form and decode information. The brain is seen as an information processor, like computer central processing, but of a different type. If so, what are the differences?
In a systems theory approach a processing system (computer or brain) is presumably composed of processors (computer or cognitive) that receive input (from sensors or ports) and create out-put (to effectors or peripherals). While the brain's design is relatively consistent between people due to genetics, a computer's design is whatever its designers choose it to be. Here, "computer" refers to computers whose design follows Von Neumann's original architecture, which is the vast majority of computers in use today. In his original design, to ensure valid processing, Von Neumann assumed:

1. **Centralized control**
2. **Sequential input**
3. **Single sourced output**

4. **Location based storage**
5. Input driven initiation
6. Minimal self-processing

Control

Centralized control means all processing ultimately originates from and returns to a central processing unit (CPU), which may delegate work to sub-processors. Computers have a CPU for control reasons, so the computer always knows exactly where, in processing terms, it is up to. A disadvantage of this architecture is that if the central unit fails, the whole system also fails.
On a software level, an operating system infinite processing loop means the whole system "hangs". Asking a room of people if their computer "hung" this week usually gives a show of hands,
especially for Windows users, but asking people if their brain "hung" this week is almost a non-question. The brain's "operating system" can work continuously for over seventy years, while the Windows operating system gets "old" after only 2-3 years, and must be reinstalled.
The human brain, unlike the computer, has no clear "CPU". In its neural hierarchy lower sub-systems report to higher ones, but the highest level of brain processing, the cortex, is divided into two hemispheres that divide up the work between them. Each hemisphere replicates its data to the other using the corpus callosum, a massive 800 million nerve fiber bridge, so both hemispheres "see" the entire visual field. Each hemisphere can independently process input and create output, i.e. it acts like **an autonomous "brain".** Subsystems for speech and memory within a hemi-sphere also have autonomy, as do psycho-motor and emotional systems.

Input

Sequential processing handles data or instructions one after another rather than simultaneously in parallel. While computers can use pipelining and hyperthreading, most computer processing is sequential due to cable and port bandwidth limits. While supercomputers have limited parallel processing, millions of human retina cells parallel process boundary contrast information before the signals leave the eye.

Parallel processing explains how the brain recognizes sentences or faces in 1/10th second, faster than most computers, yet the neuron refractory period is 1/1,000th second - a million times slower than a typical computer event. Slow brain hardware allows for only 100 sequential steps in 1/10th second, and no pro-gram can do human pattern recognition in 100 lines. The brain’s slow components create fast responses using parallel processing.

Output

Single sourced output processing “locks” output for exclusive access by one processing system, e.g. two documents sent from different computers to a network printer at the same time come out one after the other, not interleaved, as each program gets exclusive access. Databases use exclusive control to avoid the deadly embrace of a double lock. In general, computers process the same input only one way, so software updates of a program or driver overwrite previous versions.

However in the brain’s evolution, new systems overlay rather than replace older ones. The older system still remains and processes. Older systems being simpler are faster, and so can respond quicker, e.g. touching a hot stove gives an instinctive pull back. *Challenger* launches use three computers to calculate the complex “Launch” decision, and likewise the brain seems to calculate its outputs many ways, then take the best and ignore the rest. The alternative to single source output control is **overlaid output,** where different systems respond on a time gradient.

Storage

Location based storage stores, and recalls information by numbered memory locations, e.g. a disk’s side, track and sector.

While such systems can duplicate data by duplicating storage, this is costly, and so one computer “fact” is usually stored in one place, so damaging the location destroys the data held there.
Storage capacity depends linearly on the number of locations, so such systems can report “memory full”.
In contrast, brains never report a “memory full” error, even after decades of experience. One memory is not stored in one place, i.e. brains don’t store memories as computers do. That electrodes stimulating certain brain cells evoke particular memories does not make them stored at that location, just activated from there. Studies suggest that one memory involves 1,000 to 1,000,000+neurons, and one neuron contributes to many memories, rather than just one.
The brain somehow stores memory in the neural interconnections that increase non-linearly with neuron number, e.g. 1012 neurons each connected to 10,000 others gives 1016 connections - ample capacity for a lifetime’s data. This method also allows **location by content** rather than address, the computer equivalent of indexing on every field in a database record.

Initiation

The Input-Process-Output (IPO) model applies equally to the brain, whose input is the senses, and output is motor effectors.
Input driven processing means that input initiates processing which creates output, i.e. a computer’s output is defined by its input. If people worked this way, the brain would turn sensory input into behavior as a mill turns flour into wheat, i.e. mechanically. Just as without wheat there is no flour, so without sensations there should be no mental processing. Yet in sensory deprivation studies people soon start to hallucinate, i.e. the brain creates perceptions.
While computers without input typically fall “idle”, people without input get bored, and seek out stimulation. Human processing seems not a linear Input-Process-Output (IPO) sequence, but a

feedback loop modified by life, where output affect consequent input, e.g. turning the head affects what one sees.
An alternative to a mechanical input-driven system is **a process-driven system** that actively initiates feedback loop, allowing expectations, purposes and contexts.

Self-processing

The classic animal test for "self-awareness" is to show a mirror to the animal and see if they know themselves. Computers, like the less intelligent animals, fail this test. In contrast, people in-vest a great deal of time creating their "ego", or idea of them-selves, which self-concept strongly affects behavior.
While computers **do not have an "I",** or give themselves names, most people do. While programs generally avoid writing over their own code, and an operating system that overwrites itself is considered faulty, human goals like: "To be less selfish" imply people changing their own neural "programming".
Is a "self" processing itself, like a finger pointing to itself, im-possible? Not for overlaid systems, as brain systems like the frontal cortex, with autonomy, can form not only a concept of self, but also use it in social relationships.
Social Identity Theory further suggests that groups arise when individuals use a group's "identity" to form their **self- identity.**
Social development requires the ability to self-process.

Summary

In summary, the brain's design differs from that of most computers in its:

1. **Decentralized control:** The brain has no CPU, even at the highest level.
2. **Massively parallel input:** Retinal cells are massively parallel processors.
3. **Multi-sourced output:** New and old brain systems compete for output.

4. **Storage by connections:** Allows access by content and life-time's memory storage.
5. **Process driven initiation:** People plan, expect, hypothesize and predict life.
6. **Self-processing:** People have concepts of self and group that allow social activity.

The human answer to information processing is the computer, a powerful system with one central control point, that processes most input signals one at a time, that uses one program at a time for output tasks, that stores one bit of data in one place, that initiates its processing in one way, and that does not process its own processing.

In contrast, nature's solution is more subtle, with many points of control, many input signals processed at once, many output calculations per task, memories stored in many places, initiation by processing as well as input, and meta-levels of processing.

Such design differences suggest that machine intelligence and human intelligence may be as different as apples and oranges. The brain uses tactics that for computers are highly risky, e.g. self-processing risks infinite recursive loops. Yet risk also enables opportunity, so people unlike computers can think about their thinking.

While computers have people to look after them, the brain responds to undefined and potentially infinite variability in real time, where "It does not compute" is not an option.

Implications

People have always wondered how smart computers really are. As computers attempt human specialties, like walking and talking, they tend to look stupid rather than smart. As we get to know computers, their image of incredible cleverness is changing, e.g. AI enemies in computer games are considered less challenging than human opponents, hence the rise of multi-player online gaming.

While the initial “power” advances of AI were rapid, it seems now to have struck what can be called the 99% barrier, e.g. computer voice recognition is currently 99% accurate, but one error per 100 conversation words is an error per minute, which is in-adequate. There are no computer controlled “auto-drive” cars because 99% accuracy means an accident every day, again unacceptable. Real world requirements, like driving in rush hour traffic, need well above 99% driving accuracy. A competent human driver’s “mean time between accidents” is in years, not days or months, and good drivers go 10+ years with no accidents.
Human information processing has somehow crossed over into “the last percentage” of performance, but for computers more power is now giving diminishing returns, perhaps because processing power alone is not enough for “equivocal” real world tasks, e.g. pure processing logic cannot deduce a 3D world from two-dimensional retinal signals, as the brain does.
David Marr suggests that computer pixel-by-pixel processing has not led to face recognition because trying to understand perception by studying neuronal (pixel level) choices is “like trying to understand bird flight by studying only feathers. It just cannot be done.”
The solution may require not just processing power but a new processing style. Yet if computers deviate from Von-Neumann’s original deterministic assumptions, who is to say they will not then inherit human-like weaknesses?
Despite enthusiastic claims that computers will soon overtake people in intelligence technology still struggles to simulate retinal activity, let alone the visual cortex, i.e. computers struggle to simulate the lowest level of brain processing (the retina is part of the brain).
The problems of, say, computer-vision are only just beginning to be appreciated: computers are no real competition for the human brain in areas such as vision, hearing, pattern recognition and learning. And when it comes to operational efficiency there is no contest at all. A typical room-size supercomputer weights roughly

1,000 times more, occupies 10,000 times more space and consumes a million-fold more power.
Barring an unexpected breakthrough, computers in the foreseeable future will struggle with skills like conversation that five year olds find trivial, raising the question: “How long before computers learn what people learn after five?”
The Robot World Cup wants robot teams to compete with people by 2050, but the contrast between robot shuffles and world cup brilliance may be more than imagined. Perhaps the question is not whether 50 years will suffice, but whether a thousand will.

A Socio-technical Approach

The socio-technical approach gives a pragmatic model for computing progress that does not depend upon computers outperforming people in real life tasks like driving. In this view, computers are at the lowest level (hardware chips and circuits), yet software “emerges” from hardware as a new system level, based on data flow diagrams not circuit diagrams. Likewise meaning can “emerge” from data/information, and cultures from individual meanings. Higher levels offer higher performance based on higher operational requirements.
In this view, computers will progress beyond mere information processing, into areas like thinking and socializing, and soon overtake human processing, if they have not done so already. The problem with this technological utopianism is that computer AI agent development has made little progress in a decade.
In **the alternative socio-technical model,** computers combine with people to form new systems, e.g. while plane and pilot can be seen as two physical systems (human and machine) working side-by-side, or seen as one socio-technical system with human and mechanical levels (where the pilot’s body is just as physical as the plane). In the latter case the human adds a new system level not just a new system component.

In a socio-technical view, computers need not attempt tasks like abstract thinking that a perfectly good information processor (the brain) already does, perhaps as well as can be expected given the task. If people and computers are simply different types of information processors, this approach says "Vive la difference!" and aims to combine their strengths.

The goal of computing now changes, from making better computers to forming better human-computer teams, i.e. from computer excellence to human-computer excellence. The computer's role also changes, from that of clever independent actor to human assistant or social environment.

Also, socio-technical development is seen as an extension of HCI concepts. To support the view that people plus computers are more powerful that people or computers alone, note that every runaway IS/IT success of the last decade has supported rather than supplanted human activity. Computers that combine human and computer processing are succeeding more than clever systems with stand-alone processing, like the Sony dog. A Sony dog with less smarts but cuddly fur and puppy dog eyes might be more successful.

The principle of finding a human task and supporting it has many success stories, e.g. tax software, route direction systems (Mapquest), currency conversion, etc. While driverless cars are still a distant possibility, automobiles already have features like reactive cruise control, range sensing and assisted parallel parking. While computer surgery is still a dream, computer supported surgery (e.g. over distance) is a reality. While robots routinely fall over, people with robotic limbs walk well.

However, designers now have a problem beyond computer excellence, namely defining the computer/human boundary. Clever software that crosses that computer-human boundary and at-tempts human roles becomes annoying not useful. Word can be like a magic world where moved figures and titles jump about or disappear, typing "i = 0" turns into "I = 0", tables refuse to resize their columns, and text blocks suddenly change to a new format (like numbered) after a deletion.

Experienced users typically turn off auto-help options as, like the sorcerer's apprentice, software assistants soon get out of control and act beyond their ability. They act like they know better. Software that excludes the user from its operation and then acts beyond its ability is an increasing problem, not counting the programmer time wasted on functions that users turn off. Competent users dislike software that is too clever for its own good and won't listen.

Humanizing Technology

For a computer system to support human requirements they must be specified. Interfaces that work the way people work are more accepted, more effective and easier to learn. A good interface does not interrupt or annoy and remembers past user interactions.
The approach is to derive computer primitives from psychological processes, e.g. multi-media systems succeed by matching the many human senses.
It is often efficient to follow group knowledge, e.g. people often traverse a forest by following the paths trod by others. Web sites could support this social process by showing "web-tracks", i.e. changing link appearance to reflect use frequency, e.g. often-used links could increase in size or deepen in color.
Web "tracks", that show visually where other users have gone, would be highly successful. Web-trackers already gather user click data in secret for web site owner benefit. A web track site would simply make click data visible to everyone, not just a few. Of course, for web-tracks to work as a social tool, social problems would have to be overcome, e.g. a social standard would be needed to prevent sellers from abusing it to trick customers.

Conclusions

The brain is a different type of information processor, not an inferior one. This suggests replacing technological utopianism with socio-technical progress, where computers plus people form

more powerful systems than either alone. For this to occur, the computer must change its role from clever actor to simple
assistant.
It is concluded that:
1. people and computers are different types of processors, with different strengths and weaknesses
2. For computing to attempt human specialties may be not smart but dumb.
3. Computing that works with people and society will succeed more than that which tries to advance alone.
4. Human and social processes will increasingly drive computer design.
The next decade will show whether these general predictions are true or not. Perhaps in a few years robot house-help will walk in the door. Even so, the argument is not against technology centered progress, but for socio-technical combinations, i.e. why waste the processing power of the human brain?
For a global society to arise from the Internet technology it must follow rather than ignore human social guidelines.

Ex.I. Answer the following questions:

1. Why is it difficult for computers to recognize patterns?
2. The principles of traditional computer architecture according to Von Neumann.
3. What are the advantages of an autonomous brain as compared with a centralized computer control?
4. What are the advantages of parallel processing?
5. Why can't brains report a "full memory" error?
6. What are the differences between the brain's and the computer designs?
7. Are computers comparable to people when attempting human specialties?
8. What are the different socio-technical models for computing progress?
9. Can computers alone outpace people in productivity?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

Компьютеры имеют память. В нее можно записать программы, данные, изображения, что угодно. Однако, все они хранятся как некоторые именованные переменные. Специальные процедуры хеширования позволяют вычислить по имени переменной ее адрес в физической памяти, и именно по этому адресу будет искаться соответствующая запись. Никакой связи между адресом, по которому находятся данные, и содержанием самих данных не существует. Такая адресация предполагает пассивность данных в процессе поиска. Это обстоятельство чрезвычайно затрудняет поиск данных с частично известным содержанием.

Мозг использует другой способ поиска информации - не по адресу, а по содержанию, вернее, по его достаточно представительной части. Вспомните телепередачу "Угадай мелодию", в которой участникам предлагается восстановить текст куплета по нескольким нотам мелодии.

Память, способная восстанавливать полную информацию по ее достаточной части называется содержательно адресованной. При этом, мозг способен извлекать информацию и в случае, если исходные данные (ключ) являются не собственно ее частью, но связаны с ней устойчивой связью. Такая память называется в общем случае ассоциативной.

Наша память является также и распределенной. Это означает, что в мозге нет специализированного нейрона, отвечающего, например, за распознавание вашей бабушки. Наоборот, в запоминании некоторой информации участвует множество нейронов, так что гибель некоторых из них обычно не удаляет соответствующий образ из памяти. Более того, мозг обладает огромной компенсаторной способностью: поражение обширных участков приводит к тому, что соответствующие функции берут на себя другие его части. Такое свойство систем называется робастностью (robust - крепкий, здоровый). Вспомните, что произойдет с вашей программой в компьютере, если в ней запортить несколько бит - и Вы оцените достоинства хранения информации в мозге.

Распределенность не предполагает бесструктурность. Мозгу присуща определенная пространственная локализация функций. Так, в области затылка расположена зрительная кора, а лобные доли ответственны за планирование поведения.

Наконец, важнейшим свойством нашей памяти является ее активность. Суриков не видел, как Суворов переходит через Альпы, но создал в своем сознании соответствующий образ и материализовал его на холсте. Еще более причудливые примеры активности памяти могут быть найдены на картинах Босха.

Итак, человеческая память отличается от компьютерной тем, что она: содержательно-адресованная, ассоциативная, распределенная, робастная и активная.

Ex.4. Topics for discussion.

1. The computer and the brain - their similarities and differences.
2. Different ways of processing and storing information by the computer and the brain
3. The limitations of AI when attempting human specialties.
4. The future of computing is in further developing human-computer interface.

An Active Mind is the Key to Longevity

Key vocabulary:

1. Life expectancy - продолжительность жизни
2. Ageing - старение
3. Intelligence Quotient (IQ) - коэффициент интеллекта
4. IQ scores - показатели коэффициента интеллекта
5. Longevity - долголетие
6. Positron-emission tomography (PET) - позитронно-эмиссионная томография
7. Magnetic resonance imaging (MRI) - получение изображения с помощью магнитного резонанса
8. Per se (лат.) - само по себе, по сути, непосредственно
9. To live a full-fledged life - жить полноправной, самостоятельной жизнью
10. Sedentary lifestyle - сидячий образ жизни
11. Inequality - непохожесть, несходство, неодинаковость

Until recently it was widely believed that heredity, the environment and lifestyle were the main factors in human life expectancy. Today, however, there is yet another factor - the level of intellect and the intensity of intellectual activity. The link between IQ and ageing was established by specialists at the British Medical Research Council. They surveyed more than 1,000 people in different social and occupational groups, publishing their findings recently. Mortality rates among people who engage in intense intellectual activity throughout their lives are four times lower, than among those who do not. Researchers stress that a key factor here is constant intellectual activity, not just high IQ scores.

Thus far British scientists have only presented the facts but have yet to explain them. This was the question put to Academician Natalya Petrovna Bekhteeva; director for research at the Russian Academy of Sciences (RAS) Brain Institute [died in June, 2008].

Natalya Petrovna, you have often said that, like staying in good physical shape, keeping fit intellectually requires a constant effort. Now there is evidence of a link between physical and intellectual shape. What is it based on?

I can only hypothesize, based on our knowledge of human brain activity. Yevgeny Nikolaevich Sokolov, a well-known physiologist, once suggested that the brain of living beings works along "marked lines." He studied lower animals, coming to the conclusion that their brain activities were preprogrammed. Their neuron network is so formed that nerve impulses in the brain spread like trains along rail tracks with prearranged stops. The human brain is far more complex, but even ***we*** have such "marked lines." This is how our accustomed, automatic actions proceed - e.g., when entering a room, you automatically switch on the light.

Scientists can observe what happens in the brain when a person performs different actions - both simple and complex. When a person begins to do something new, at first the entire brain is activated, but gradually some of its fields are turned off with only those functioning that are necessary for a particular type of activity. The simpler an activity is, the fewer brain fields are "on." This saving mode frees up the brain for something bigger.

What bigger things?

Complex intellectual activity, solution of nonstandard problems and creativity. Recent research, based on the use of positron-emission tomography (PET) and magnetic resonance imaging (MRI), showed that, in these cases, certain sections are activated in many fields of the brain. For the last five years, the RAS Brain Institute has been studying the organization of the brain in the process of creative activity. Volunteers are asked to perform tasks of different complexity (e.g., compose a story from connected and disconnected words), and then PET and MRI are used to observe what is happening in the brain at this moment. It turned out that creative activity taps almost the entire brain, including zones that are associated with very different processes - memory, emotions, creativity per se, and many other things.

But how is all of this related to a person’s physical shape and life expectancy?

With such “unconventional activity” the brain is used to the maximum, living a full-fledged life. Like other organs, the brain needs work. Even the brain of children who do not receive sufficient stimulation may not develop properly. If however, a person has spent his life in a “stereotype” conventional routine situation - sweeping the roads, working on the assembly line, etc.

- in other words, moving along “marked lines” without any compensation for this underemployment of the brain, in old age he will not be able to study a foreign language, will have serious memory problems, and so on; On the other hand, a person who has engaged in intensive intellectual activity pre-serves his brain in good working order until a very old age, even though his memory may not be as good as at age 20 or 30.

But another thing is very important. In fulfilling “super-tasks,” the brain of even a not very young person can create new connections and even form new neurons and nerve cells. Emerging neuron networks begin to work and contribute not only to good intellectual shape but also influence many processes in the organism. An active brain can better cope even with the effects of a stroke.

Super-tasks are useful, even vital for the human brain. Compare a large city with many people constantly moving around, extensive infrastructure, and plenty of lights, on the one hand, and an abandoned village where a handful of elderly people are living out their time, on the other. A human brain that does not solve complex tasks and does not engage in creative activity is like an abandoned village.

What about the body?

The brain has to do with everything that is happening in the organism. For example, creative thinking activates zones that are essential for emotional activity, including hypothalamus structures influencing the endocrine system which is directly related to ageing processes. This is only one line, but I believe that an active

brain in elderly age continues to regulate almost everything that happens in the organism.
I have devoted much thinking to the central regulation of the ageing process: There are several phases of ageing, and it is quite possible that the brain can extend one of these phases.
Why has intellectual activity not been seen as a factor in life expectancy?
You see, the brain can do very much, but not everything. Often times "intellectual workers" or "brain workers" lead an unhealthy lifestyle - low physical activity, a sedentary lifestyle, wrong diet, bad eating habits, and bad habits in general. The brain at-tempts to compensate for this, but cannot always cope with the problem. Although these people live longer, they are unhealthy. My judgment is based both on my own experience and the people around. So the best possible option is to combine physical and intellectual activity from an early age. "Brain workers" past 40 should give this some thought.
What about others? Can something be changed after 40?
Some of my patients had memory problems, and I asked them: "Do you read much?" "Yes, we read newspapers," they would say. At that time, our newspapers almost did not differ from each other, so I would tell them: "Unless you start reading something else, I do not envy your old age." People should at least read different literature. It is essential, as far as possible, to apply unconventional approaches to various problems. There are many things in a person's life that can stimulate the brain, challenge and force it to come up with unconventional solutions, even on the day-to-day level, within the family.
Unfortunately, this approach cannot be imposed on a person. But on the other hand, organized propaganda and advertising can be rather effective. Consider the popularity of fitness clubs today. It might be a good idea to organize "brain clubs".
Do you have some personal "intellectual health" rules?
I learned one rule from my boss, Dmitry Andreevich Biryukov, director of the Institute of Experimental Medicine, whose working

day was sharply divided into two halves: during the first half he devoted himself to science and during the second, to management. When he was in the laboratory, nothing (unless it was a fire) could distract him from his research. When I became director, I did the same, but was unable to pursue full-fledged research for very long.

Today, I work most of the time at home, and this requires very strict discipline. Until I have produced my "daily quota of pages" or until I have read what I have assigned myself to read for my current project, I do not deal with the so-called chores which can consume much time and effort.

FACT BOX 1

The Brain and the Computer

The human brain contains 10 billion to 20 billion neurons, nerve cells. One neuron has links with about 20 000 other neurons. It takes a few microseconds to transmit chemical signals between neurons, while the brain performs many functions farter than the most powerful modem computers. The trick is that the brain does not follow encoded instructions but activates links (synapses) between the neurons. Each activation is equal to the performance of a specific digital command. The brain's synaptic activity can be as high as 1016 neuron connections per second. To achieve this productivity, it will take 1 million Intel Pentium computers, consuming a total of hundreds of mega-watts.

FACT BOX 2

IQ linked to health

Intelligence may partly explain why some people enjoy worse health than others - but it is not the whole story, researchers say. Some experts have suggested that IQ is an important factor in explaining the wide health inequalities which exist in the UK and elsewhere.

But strong evidence of this link is scarce, with few studies conducted in efforts to confirm the association and use this information to reduce the health gap.
Now researchers from the Medical Research Council, writing online in the British Medical Journal, have looked at the effect of IQ on the health of 1347 men and women living in the west of Scotland.
The participants, who were aged 56 when the study started in 1987, had their IQ assessed in a written test, while their socio-economic status was identified by interview.
Their health was then monitored for 17 years by researchers from Glasgow University and Edinburgh University.
The researchers wanted to discover whether taking account of IQ when looking at health would cause the link between the socioeconomic position and health to disappear.
They assessed factors such as heart disease deaths, long-term illness and respiratory function.
As was expected, those in the poorest groups faced the greatest link of ill-health and death.
When they took into account IQ, the link between social group and ill-health was markedly reduced.
But the risk of ill-health among disadvantaged people was still twice that of the more advantaged group with many of the health factors looked at in the study.
"Our findings indicate that IQ does not completely explain the health differences between rich and poor, but may contribute to them," the researchers said.
One explanation for the link between the intelligence and health inequalities may be that IQ is linked with behavior which can lead to ill-health.
The researchers said, for example, that people with higher child-hood IQ scores were more likely to stop smoking once started than those with lower IQ scores.
They also said that important chronic diseases, such as hypertension and diabetes, had been shown to lower IQ.

Ex.l. Answer the following questions:

1. What is the link between IQ and ageing?
2. What is this link based on?
3. Comment on the difference between the neuron network of lower animals and human brain activity.
4. Describe the brain activity when a person performs different actions.
5. How does fulfilling super-tasks stimulate brain activity?
6. Why does intellectual activity help to preserve one's brain in good working order until a very old age?
7. Why is a human brain that does not solve complex tasks and does not engage in creative activity compared to an abandoned village?
8. How is the brain activity connected with ageing?
9. What can stimulate the brain activity?
10. What do the brain and computer activities have in common?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

Исторически сложились три основных направления в моделировании ИИ.
В рамках первого подхода объектом исследований являются структура и механизмы работы мозга человека, а конечная цель заключается в раскрытии тайн мышления.
Второй подход в качестве объекта исследования рассматривает ИИ. Здесь речь идет о моделировании интеллектуальной деятельности с помощью вычислительных машин. Целью работ в этом направлении является создание алгоритмического и программного обеспечения вычислительных машин, позволяющего решать интеллектуальные задачи не хуже человека.

Наконец, третий подход ориентирован на создание смешанных человеко-машинных, или интерактивных интеллектуальных систем, на симбиоз возможностей естественного и искусственного интеллекта.
Самыми первыми интеллектуальными задачами, которые стали решаться при помощи ЭВМ были логические игры (шашки, шахматы), доказательство теорем. Хотя, здесь надо отметить еще кибернетические игрушки типа "электронной мыши" Клода Шеннона, которая управлялась сложной релейной схемой. Эта мышка могла "исследовать" лабиринт, и находить выход из него. А, помещенная в уже известный ей лабиринт, она не искала выход, а сразу же, не заглядывая в тупиковые ходы, выходила из лабиринта.
Американский кибернетик А. Самуэль составил для вычислительной машины программу, которая позволяет ей играть в шашки, причем в ходе игры машина обучается, улучшая свою игру на основе накопленного опыта. В 1962 г. эта программа сразилась с Р. Нили, сильнейшим шашистом в США, и победила.
Ярким примером сложной интеллектуальной игры до недавнего времени являлись шахматы. В 1974 г. состоялся международный шахматный турнир машин, снабженных соответствующими программами. Как известно, победу на этом турнире одержала советская машина с шахматной программой "Каисса".
В настоящее время существуют и успешно применяются программы, позволяющие машинам играть в деловые или военные игры, имеющие большое прикладное значение. Одной из наиболее интересных интеллектуальных задач, также имеющей огромное прикладное значение, является задача обучения распознавания образов и ситуаций. Интерес к этой задаче стимулировался фантастическими перспективами: читающие автоматы, системы ИИ, ставящие медицинские диагнозы, проводящие криминалистическую экспертизу

и т. п., а также роботы, способные распознавать и анализировать сложные сенсорные ситуации.
В 1957 г. американский физиолог Ф. Розенблатт предложил модель зрительного восприятия и распознавания — перцептрон. Появление машины, способной обучаться понятиям и распознавать предъявляемые объекты, оказалось чрезвычайно интересным не только физиологам, но и представителям других областей знания и породило большой поток теоретических и экспериментальных исследований.
Проблема обучения распознаванию тесно связана с другой интеллектуальной задачей - проблемой перевода с одного языка на другой, а также обучения машины языку. При достаточно формальной обработке и классификации основных грамматических правил и приемов пользования словарем можно создать вполне удовлетворительный алгоритм для перевода, скажем, научного или делового текста. Для некоторых языков такие системы были созданы еще в конце 60х г. Однако, для того, чтобы связно перевести достаточно большой разговорный текст, необходимо понимать его смысл. Работы над такими программами ведутся уже давно, но до полного успеха еще далеко. Имеются также программы, обеспечивающие диалог между человеком и машиной на урезанном естественном языке.

Ex.4. Topics for discussion.

1. Human life expectancy and the level of intellectual activity.
2. The level of IQ and its influence on health inequalities.
3. The brain’s productivity vs. the computer productivity.

Nanotechnology -the Miracle of the 21st Century?

Key vocabulary:

1. Scanning probe electron microscope (SPM) - сканирующий электронный микроскоп
2. Top down - нисходящий, выполняемый сверху вниз
3. Bottom up - восходящий, выполняемый снизу вверх
4. Self assembly - самосборка
5. DNA molecules - молекулы ДНК
6. Coating - покрытие
7. Self-replicating - самовоспроизводящийся
8. Nano-particles - наночастицы
9. To leapfrog - продвинуться вперед, перепрыгнуть
10. Optical fibre - оптоволокно
11. Breakthrough - прорыв

The term ‘nanotechnology’ encompasses a huge range of activities. ‘Nano’ is used in the world of science to mean one billionth. E.g. **a nanometer** is a billionth of metre. A nanometer is only ten atoms across! So generally nanotechnology is used to mean technology at the nanometer level. Nanotechnology attempts to achieve something useful through the manipulation of matter at this level.
To put it more formally, you can use the following definition:
“Nanotechnologies are the design, characterization, production and application of structures, devices and systems by controlling shape and size at nanometer scale”. At such scales, the ordinary rules of physics and chemistry no longer apply. For instance, materials’ characteristics, such as their colour, strength, conductivity and reactivity, can differ substantially between the nanoscale and the macro. Carbon ‘nanotubes’ are 100 times stronger than steel but 6 times lighter.

History. Physicist Richard Feynman gave a lecture to the American Physical Society in 1959 which foresaw advantages from
manufacturing on a very small scale - e.g. in integrated circuits for computers, for sequencing genes by reading DNA molecules and using machines to make other machines with increasing precision. However, the term 'nanotechnology' was first used by Norio Taniguchi in 1974, in a talk about how the accuracy of manufacturing had improved over time. He referred to 'nanotechnology' as that which achieved greater dimensional accuracy than 100nm.
Feynman also envisaged machines that could pick up and place individual atoms. This development of this idea was later assisted by the invention of the scanning probe electron microscope (SPM) which allowed scientists to 'see' and manipulate the individual atoms in a surface. In 1989 one of the defining moments in nanotechnology occurred when Don Eigler used an SPM to spell out the letters IBM in xenon atoms. For the first time scientists could put atoms exactly they wanted them.
Molecular building blocks. Another great leap forward occur-ed in the shape of a new form of carbon. Harry Kroto from the University of Sussex, together with Richard Smalley and Robert Curl, discovered the carbon 60 molecule, which is shaped like a soccer ball. They named the molecular structure after the similarly shaped geodesic dome structure pioneered by the architect Buckminster Fuller. Unfortunately, 'Buckminsterfullerene' is too long a name for most people and so they are often called 'bucky-balls'.
There are two fundamentally different approaches to nanotechnology, termed 'top down' and 'bottom up'. **'Top-down'** nanotechnology features the use of micro- and nanolithography and etching. Here, small features are made by starting with larger materials (e.g. semi-conductors) and patterning and 'carving
down' to make nanoscale structures in precise patterns. Complex structures including microprocessors containing hundreds of mil-

lions of precisely positioned nanostructures can be fabricated. Of all forms of nanotechnology, this is the most well established.
‘Bottom-up’, or molecular nanotechnology (MNT), applies to building organic and inorganic structures atom-by-atom, or molecule-by-molecule. Here we are using the forces of nature to assemble nanostructures - the term ‘self assembly’ is often used.
The self assembling properties of biological systems, such as DNA molecules, can be used to control the organization of species such as carbon nanotubes, which may ultimately lead to the ability to ‘grow’ parts of an integrated circuit, rather than having to rely upon expensive ‘top-down’ techniques. Nanotechnologies are widely seen as having huge potential in areas as diverse as healthcare, IT and energy storage. Governments and businesses across the world have started to invest substantially in their development.
However, there are also concerns regarding the safety of nanotechnology. These range from the more fanciful (such as Eric Drexler’s imagined scenario of a world reduced to ‘grey goo’, caused by self-replicating nano-robots) to the more realistic (such as the possible dangers of foreign nano-particles entering human organs and the bloodstream).
Some short-term nano uses: Medical diagnostic tools and sensors. Solar energy collection (photovoltaics). Direct hydrogen production. Flexible display technologies and e-paper. Compo-sites containing nanotubes. Glues, paints and lubricants. Hew forms of computer memory. Printable electronic circuits. Various optical components.
Some longer-term nano uses: Miniaturized data storage systems with capacities comparable to whole libraries’ stocks. PCs with power of today’s computer centres. Chips that contain movies with more than 1,000 hours of playing time. Replacements for human tissues and organs. Cheap hydrogen storage possibilities for a

regenerative energy economy. Lightweight plastic windows with hard transparent protective layers.
Ever since John Daiton convinced the world of the existence of atoms in 1803, scientists have wanted to do things with them. Nanotechnology takes that ability on to a new plane and opens up all kinds of futuristic imaginings. Essentially, nanotech is the manipulation on the molecular scale - distances that may cover just a few millionths of a millimeter. But its potential is not just about being able to miniaturize things. Indeed, scientists and engineers recognize that there are fundamental limits to pure miniaturization. Working at a scale a million times smaller than a pinhead allows researchers to 'tune' material properties, making them behave in different ways to normal, large-scale solids. This behavior can be exploited in quite ground-breaking ways.
Nature has been doing nanotechnology for a long time, and it has become expert in it. Consider the super-fine hairs on a gecko's feet which allow it to stick to walls and even hang upside down on a glass sheet. Learning from nature, nanotechnology promises humans ways of making systems that are smaller, lighter, stronger, more efficient, but cheaper to produce. "Nanotechnology is not a technology in its own right," explained Professor Mark Welland, head of the University of Cambridge Nanoscale Science Laboratory. "It is an enabling technology, so it will appear in many different products. It is already appearing in flash memory, computer chips, and it will increasingly be an enabling technology in other products like coatings, new types of sensors, especially in the medical area."
It is expected to transform the performance of materials, like polymers, electronics, paints, batteries, sensors, fuel cells, solar cells, coatings, computers and display systems. In five years' time, batteries that only last three days will be laughable, said Professor Welland. Similarly, in 10 years' time, the way medical testing is done now will be considered crude. To say that in five years an

iPod will have 10 times its current storage capacity will be conservative, he said. In the not-so-distant future, a terabit of data - equivalent to 10 hours of fine quality uncompressed video - will be stored on an area the size of a postage stamp. Clearly, the devices themselves will not be nanosized. But nanotechnology will play its part in shrinking components, and making them work together a lot more efficiently. Although nanodevices can be built atom by atom, it is not realistic as a manufacturing option because it is slow and expensive, thinks Professor John Ryan, head of the Bionanotechnology Center at Oxford University. “One of the major scientific challenges in the years ahead is to understand the fundamental biological principles and apply them to produce new types of nanotechnology,” he said. “Armed with these design rules it may then be possible to make new types of nano-device using materials that are more robust than bio-materials.”

The Royal Society and the Royal Academy of Engineering has looked at current and future developments in nanotechnology and has reported on whether it will require new controls. It is hoped that the report grounds some unrealistic scenarios, while recognizing that real concerns need to be addressed with regulation. “The one fantastic idea that has dogged nanotechnology is the self-replicating machine, the ‘grey goo’, scenario,” said Professor Welland. “That is simply too far off. The complexity of designing a molecular machine is bad enough, but if you try to imbue that with self-replication, you could not even put a toe in the water to design it.” The scenario sees swarms of self-replicating robots, smaller than viruses, multiplying uncontrollably and devouring the Earth. Eric Drexler, who many consider to be ‘a farther of nanotechnology’, has distanced himself from the idea, saying such self-replicating nanomachines are unlikely to be widespread. Similarly, fears over ‘grey goo’, the concern that self-replicating, nano-sized biological particles will move into human

bodies and do unpredictable things, is scaremongering, thinks Professor Welland.
Professor Ryan agrees: “These science fiction scenarios have not only diverted attention away from the real advantages of nano-technology, but also from issues that do raise concern.” Inhaled nano-particles found in the bloodstream which have dispersed throughout the brain is a concern, he says. Whether this poses a health risk is not known. “If you look around at the moment in a big city, a significant proportion of material that you breathe in is already particulates - and a proportion of that is nano-sized, like diesel emissions,” said Professor Welland.
Nano-materials exploit unusual electrical, optical and other properties because of the very precise way in which their atoms are arranged. This means fabrics could change colour electronically.
Exposing an army uniform to ultra-violet light could activate changes without undressing. But it is in medicine that nanotechnology offers the most remarkable advances, according to Professor Ryan. “Nanomedicine will provide earlier and better diagnostics and treatment will combine earlier and more precise-lytargeted drug delivery,” he said. The possibility of individualized therapy is also on the horizon. Nanotechnology in the form of flexible films containing miniaturized electrodes is expected to improve the performance of retinal, cochlear and neural implants. And it could lead to the miniaturization of medical diagnostic and sensing tools which could drive down costs of such kits for developing countries. In this respect, nanotechnology could enable developing nations to leapfrog older technologies, in the way that copper wire and optical fibre telephony were superseded by mobile phones.
Industrial giants like GE are heavily involved in developing nanotechnology. “We think that the biggest breakthroughs in nanotechnology are going to be in the new materials that are developed,” said Troy Kirkpatrick at GE Global Research. These

include corrosion-resistant coatings to make hydro-electric turbines more efficient in heavily-silted waters, and nano-membrane water filters to make for faster filtration. GE is also studying the properties of nano-ceramics, which can offer extreme strength, while still being lightweight. Because of the molecular structure of such materials, nano-ceramic coatings on aircraft could make them 10% more efficient, so less energy is used, producing fewer emissions.
GE Global Research is also looking to the electronics industry. “If you look at the chip makers of the world, the challenge they have is not to figure out how to make them faster. The problem is they run so fast, the chips generate too much heat and melt. They need better materials for heat management,” said Mr. Kirkpatrick. Using materials which exploit the properties of nano-particles, GE has developed chip adhesives that can transfer heat out of the processor system more efficiently. “It is a start, and it is to show nanotechnology is finding its way into production and is changing the way we are doing science,” said Mr. Kirkpatrick.
Whatever nanotechnology does for the future, it will be an evolutionary process. One certainty is that there remains a plethora of uncertainties in the emerging field of nanotechnology. “Medical sensing is very attractive to everybody, but there could be a downside,” explained Professor Welland. “If medical sensors become ubiquitous, our physical state could be monitored 24 hours a day, and if someone hacked into that data, there could be concerns.” Which is indeed why regulation has to be addressed, but must not stifle nanotechnology’s potential. “One of the important things for me is that it ultimately means the most efficient use of materials and processes, which means it does not have to benefit just the G8 nations,” argued Professor Welland. “These sorts of materials, if they are able to do their job using less energy, should be available to everybody.”

Ex.l. Answer the following questions:

1. Give the definition of nanotechnology.
2. How can materials' characteristics change on nano-scale?
3. What is the difference between 'top-down' and 'bottom-up' approaches in nanotechnology?
4. Why should 'top-down' technique be more expensive than 'bottom-up' one?
5. What are the short-term and long-term nano uses?
6. How does nanotechnology reveal itself in electronics?
7. What can be the dangers of the uncontrollable development of nanotechnology?
8. What are the benefits and the downside of medical sensing?

Ex. 2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

Согласно определению, к нанотехнологии "относят два типа устройств: те, которые строятся bottom-up (из молекулярных компонентов, которые затем собираются по принципу молекулярного распознавания), и те, которые строятся top-down (из более крупных объектов без контроля на атомном уровне), имеющих размеры между одним и ста нанометрами." Однако, определение научно-технологической дисциплины через возможность постройки прибора и его размер представляется неадекватным масштабу заявленной при этом проблемы и инвестициям, исчисляемым миллиардами долларов.

К нанотехнологиям относят самые разнообразные устройства и области знаний, Не слишком ли широко? Ведь это приблизительно то же самое, что назвать килотехнологиями любые устройства, масса которых превышает один килограмм. При этом стратегическая перспективность для цивилизации создания устройств, функционирование которых определяется в субатомных масштабах, сомнения не вызывает. Надо лишь правильно сформулировать приоритеты и ключевые проблемы.

Нанопроцессы разумно подразделить на а) такие, которые происходят не только в нано-, но и в больших пространственных масштабах, и б) такие, для которых наличие нано-размеров принципиально существенно. Представляется целесообразным процессам, которые происходят только в наномасштабах, дать особое наименование. Название ‘нано-фундаментальные процессы’ представляется наиболее адекватным.

В этой связи кажутся правомерными и актуальными вопросы:

1. какие физические процессы, которые могут быть использованы при создании нанотехнологических материалов и устройств, могут существовать только в наномасштабах;
2. какие именно технологии могут быть созданы на основе нанофундаментальных процессов, для которых пространственный масштаб, сравнимый с размерами атомов и молекул, принципиально существенен?

Ответ на эти вопросы определяет стратегические направления развития нанотехнологий, а в более широком контексте - и цивилизации XXI века вообще.

Ex.4. Topics for discussion.

1. The advantages and disadvantages of both ‘top-down’ and ‘bottom-up’ approaches in nanotechnology.
2. The application of nanotechnology in different spheres of human life.
3. Miniaturization - its pros and cons.
4. The nanotechnology’s potential.

Computer Graphics

Key vocabulary:

1. Visualization technologies - технология визуализации
2. Human interface - интерфейс с пользователем
3. Distributed ambient computing - распределенные общие вычисления
4. Next-generation communication technologies - коммуникационные технологии следующего поколения
5. Sensing architecture - архитектура восприятия, считывания
6. User-interface-design - проект, разработка пользовательского интерфейса
7. Ubiquitous mobile communication networks - распространенные, повсеместно встречающиеся мобильные сетевые коммуникации
8. Information Society Technologies (IST) - технологии информационного общества
9. Information and Communication Technologies (ICT) - информационные и телекоммуникационные технологии
10. Frontier Enabling Technologies (FET) - технологии, открывающие новые области
11. Image-based rendering (IBR) - рендеринг, визуализация, основанная на анализе изображения
12. Frame grabber - устройство захвата изображения, устройство ввода и регистрации кадров
13. Embedded - вложенный, встроенный
14. Bandwidth - пропускная способность, ширина спектра, сигнала
15. Phalanx - фаланга, группа, община
16. Visualizer - визуализатор, блок видеоконтроля
17. Salient information - существенная информация
18. Automatic restoration (AR) - автоматическое восстановление

19. Acquire on the fly - приобретать на лету
20. Graphics engine - графический процессор, графическое ядро

Grand Challenges in the Evolution of the Information Society

The Information Society Technologies Advisory Group report “Grand Challenges in the Evolution of the Information Society” describes the grand challenges of the Information Society Technologies (IST) Program of the European Union and reports on some of the enabling technologies, research challenges, and societal innovations required to make those grand challenges happen. Needed enabling technologies and research challenges are: visualization technologies, cognitive technologies, human interfaces, distributed ambient computing (ambient intelligence), and advanced knowledge management.

Needed innovative Information and Communication Technologies (ICT) methods and infrastructures are: software-intensive system development; system modeling and simulation; simulated realities; next-generation communication technologies; nanoelectronics and sensing architectures.

Needed societal innovations and socioeconomic challenges (especially, from the point of view of the European Union) are: accelerating economic growth and job creation by new value and services; dealing with health care in an ageing society (ambient assisted living); and giving priority to trust, security, and privacy in ubiquitous mobile communication networks.

Looking at these enabling technologies and research challenges and considering the Frontier Enabling Technologies (FET) re-port, it’s obvious that a major need for computer graphics has been clearly identified.

Computer graphics in this context is defined as visualization technologies for presenting data, information, and knowledge;

visual, multimedia, and multimodal interaction techniques; and visual communication in ubiquitous mobile communication networks.
Computer graphics (identified as an enabling technology to make those grand challenges of the IT society happen) will play a major strategic role in future IST research and development. Computer graphics will have a similar importance and relevance in, for example, nanotechnologies, embedded systems, and sensor technologies and networks. To fulfill and satisfy such an enabling role, the computer graphics community has to address and solve some visions and challenges that the ICT world is facing.
Image-based rendering (IBR) has been an active research topic for more than a decade. The basic idea of IBR is to use a set of given images from a scene and to synthesize novel views of this scene from these images. While humans can easily imagine what the scene would look like from somewhere else, the basic algorithms proposed so far still have problems dealing with this task efficiently.
Due to the high computational costs, the state of the art is strictly limited to offline systems — most systems focus on either static scenes or are limited to static camera setups. Similar problems arise with sound, an essential part of multimedia: a complete environmental modeling includes not only 3D objects but also 3D sound showing characteristics such as directionality and space mood. A multimedia experience where users feel a sense of being present in a virtual environment requires correct modeling of both aspects.
For a number of applications, we would want to synthesize novel views from a set of video cameras in real-time video for interactive systems. Examples include the following:
Interactive TV. Using multiple channels for digital television, the spectator could choose independently of the cameraman which part of a sports event they will look at and from which viewpoint. Similar applications demanding a virtual-being-there feeling include teleconferencing and virtual walkthroughs.

Remote tower. Smaller remote airports often cannot afford their own air traffic controller; large airports often cannot be visually controlled from a single viewpoint on the tower. Both situations require camera-based solutions for virtually being there in high quality and in real time.
Surveillance. Large sites require a large number of cameras for complete surveying. It's difficult for the operator to gain an overview of the site from the native camera images only. The challenge of real-time IBR consists of three major difficulties:
Bandwidth. While it's already difficult to capture multiple, synchronized, high-resolution video streams, the idea of processing these streams in real time is nowadays impossible due to bandwidth limitations in the frame-grabber hardware, the computer, and (in parallel processing) the distributed computing environment (network).
Image analysis. An important input for the synthesis of novel views is an appropriate estimation of depth. Techniques for the passive estimation of depth from multiple views are still subject to active research. Much work still needs to be done to get sufficient quality in high resolution and in real time.
Image synthesis. While the problem of image quality has been successfully addressed, researchers still need to devise how to compactly represent the relevant information such that the final image synthesis can occur with low computational effort on end-user commodity equipment, for example, on advanced TV set-top boxes.

Smart Interface Architectures

With the advent of ubiquitous computing environments and the associated evolution of devices for information display, the risk for inappropriate interface design increases. While consistently good and task-based interface design has already been an unsolved challenge for the traditional windows, icons, menus, pointing devices interfaces since their inception in the 1960s, the introduction of a phalanx of devices with varying screen sizes, screen resolutions, modalities, and interaction affordances has increased the problem of ensuring appropriate interface design for the end user. Moreover, the breadth of devices, tasks, and environments that the modern end user might be associated with when interfacing with a computing system makes it literally impossible from the point of view of interface designers to anticipate the uses and users ahead of time.

The end user suddenly becomes the only component of the user-interface design process that has all the information necessary to specify the requirements needed for the user interface. This circumstance requires a change of interface designer and end-user roles, where interface designers create interface components out of context for specific use cases and describe them in terms that end users can query for their own purposes.

At the same time, the end users need to be empowered to efficiently design (or customize) appropriate and consistent interfaces based on preexistent use cases and the information on the context currently at hand. That means that the design knowledge needs to be captured and provided for in the form of rules, guiding the user-interface-design illiterate end users in adapting their interface to the computing environment, even eventually performing some adaptations automatically.

A prerequisite for such support is research and development in a variety of user-interface disciplines, with a strong need for cross-discipline collaboration. Needed are intelligent interface objects that—based on user-interface designers' expertise and guidelines—know how to present themselves most appropriately in different contexts and interaction modalities.

Since display and interaction devices will continue to evolve, corresponding frameworks and architectures have to be established to ensure adaptation and customization to, as well as interoperability between, different display and interaction platforms. Such frameworks and architectures have to further support

the provision of extensible interface object repositories. Such repositories must suit specific application domains as well as cross-application domains, and also allow for selective access based on user expertise and access privileges.
For all such repositories to interoperate, we need international standardization initiatives that define the metadata and describe interface objects in a cross-cultural and cross-application oriented way.

Personal Everywhere Visualizer

Our information society is facing a continuous growth of data at exponential rates. Often, however, we get overloaded with use-less data; it's difficult for us to extract relevant information. This information overload dramatically affects many aspects of our professional and private lives.
Some relief is provided by methods for interactive visualization and display, which offer an effective way to access, process, and filter information. These methods, however, require high-resolution computer screens or sophisticated immersive projection technology to fully unfold their potential, which makes them impractical for mobile use.
In a highly mobile society, personal and professional success will depend tremendously on instant availability and visualization of salient information. Users must be able to access everywhere and at all times such interactive information display and it must adapt to the individual user's needs and demands.
Recent advances in information and communication technology—in particular, mobile, wireless communication, and advanced display—provide enabling technologies to meet this challenge. This vision might be called the personal everywhere visualizer (PEV). For instance, a mobile user will employ the PEV to create a telepresence experience. The built-in projection optics will generate an instant, high-quality display and visualize a remote

communication partner on an arbitrary surface. At the same time, the device will capture high-resolution video of the user and the environment and transmit it to the remote site. In such a scenario, the PEV will constitute an enabling technology for instant presence and collaboration.
In a mobile virtual office scenario, the PEV will create an AR office environment by overlaying synthetic imagery onto a real scene. The user will have full access to all relevant information and documents as if he or she was in a physical office, while advanced retrieval and visualization methods will support efficient search and document processing. Novel, multimodal inter-action metaphors will allow the user to interact with a virtual mobile office. In idle mode, the PEV could continuously acquire and refine information about the actual environment to optimize display quality.
Another potential application is in medicine. Not only would the PEV provide instant communication and diagnosis by remotely located experts, but it could also instantly display salient medical information about the patient. For instance, a surgeon might use the PEV in the operating room to overlay high-quality x-ray images or other information onto the patient's body. Likewise, real-time image capture would acquire patient data on the fly.
A further potential application is the intelligent retail store. Here, customers could use the PEV to identify products and optimize the shopping experience. For example, a customer's handheld PEV would overlay high-quality projection onto retail shelves thus creating an AR scenario. The PEV could display information about products, for example, and enable customers to inter-act with them.
Similarly, the PEV potential for AR could be harvested for tele-learning. For instance, museum visitors could connect their PEV to the museum database. On demand, the PEV would display background information onto any art object or nearby wall. This information could be customized to the visitor's knowledge and preferences.

The creation of such a device requires powerful information and communication technology. A large number of fundamental re-search questions have to be addressed in the fields of computer graphics, computer vision, visualization, animation, imaging, multimodal interfaces, display technology, and mobile communications. For instance, we will have to investigate advanced display technology, such as digital light processors, autostereo-scopic displays, laser displays, holographic displays, and so on. Of research interest are also combinations of intelligent projection technology with 3D dynamic shape and appearance acquisition. Various types of miniature cameras and other sensors must be combined and studied. We need novel methods to render and visualize abstract information and to composite them into real environments. In particular, the interaction between rendering and data capture must be investigated. High-quality light simulation and light source estimation are further issues. Research must be devoted to information representation, coding and compression of information, decentralized storage, and streaming. It is quite probable that the PEV will eventually let the human-computer interface disappear. To this end, we have to pursue fundamental research in multimodal interaction. Other important research issues include system software, application programming, intelligent data mining, as well as perceptual and usability studies.
The PEV will let users conveniently adapt the presented information to their individual needs. This requires extensive research in authoring tools. For automatic methods, recent advances in machine learning bear great potential.

Paintable Computing

Consider the scaling limitation for large area, real time display systems. For systems based on the generic architecture (when process and data flow in display systems and image data from multiple sources converges on a single graphics engine for de-

compression, color space conversion, rendering, formatting and scaling and the graphics engine passes rasterized data through high bandwidth channel to the display), data from a variety of external image sources (both natural and synthetic) must first pass through a graphics engine for rendering and formatting.
The rendered output is then channeled to a display component for distribution to individually addressable display elements (pixels). Efforts to scale up the number of pixels ultimately confront a number of obstacles, both quantifiable engineering hurdles and subtler impediments to acceptance by the end users. Engineering limitations are typically caused by shared resources that bottle-neck; ***e.g.*** graphics engines with finite aggregate compute capacity, and data buses with bounded transmission bandwidth. Manufacturing processes for the displays likewise have difficulty maintaining adequate yield as the number of pixels per display increases. For the end customer, ultra high resolution wide area displays must be treated as fixed objects that are mechanically and electrically sensitive, require substantial infrastructure, are often bulky, are always complex, and are difficult to reconfigure opportunistically.
The last decade has seen dramatic headway made on a portion of this problem space - namely that of the display component. Emblematic of this is the work on electrophoretic ink (e-ink) where advances in the production and handling of microcapsules have yielded bi-stable, printable displays with the viewing affordances of paper and refresh rates nearing those of early CRT's (cathode-ray tube).

Nevertheless, progress on the display component has only under-scored the unmet challenges at the system's level. Design of backend systems with sufficient rendering power and transmission capacity to feed a 109 pixel display in real time still outpaces today's best engineering practice. Contemporary solutions adopt the approach of tiling the active display area among autonomous display systems operating in synchrony, much in the spirit of early systems that rack mounted multiple TV's into a rectangular grid. There is a simple approach to a novel extreme. It is assigning a full-featured graphics system to every pixel.

Core to this approach is the architectural work on "paintable computing" and its associated programming methodology based on informational self-assembly.

Architecturally, paintable computers are defined as **agglomerate of numerous, finely dispersed, ultra miniaturized computing particles; each positioned randomly, running asynchronously and communicating locally.** Individual computing nodes are vanishingly cheap, freely expendable, and consequently are handled as bulk.

The task is to work on applying the architecture and programming model of **paint** to the design of display systems. There arises the question: "Can the display systems themselves become paintable?" The goal is to create display systems with the same characteristics we target for **paint;** scalability, ability to reconfigure, resilience to fault, and self-repair.

Background - Paintable Computing

Work on paintable computing pursues the vision of a machine consisting of thousands of sand-grain sized processing nodes, each fitted with modest amounts of processing and memory, positioned randomly and communicating locally. Individually, the nodes are resource poor, vanishingly cheap and regarded as freely expendable. Yet, when mixed together they cooperate to form a machine whose aggregate compute capacity grows with the addition of more nodes. The ultimate goal is to recast computing as a particulate additive to ordinary materials such as cloth, furniture, building materials or paint.

Hardware Architecture

The atomic element of a **paintable** is the particle. Characteristic specs include a ’486 class micro, an internal clock running at ~ 200 MHz, and 250K-1M of RAM for code and data storage. All the I/O to the micro is gated through a wireless transceiver supporting a full duplex rate in the range of 1-20 Mb/s. Communication is via asynchronous links to the nearest neighbors. A power subsystem harvests power from the immediate environment with minimal constraints on the particle’s placement. Once exposed to power, each particle builds an enumerated list of the neighbors with which it can communicate.

The programming model is based on the concept of process self-assembly — the unsupervised re-assembly **of a running process from fragments of code that are mobile in a virtual environment.** Process self-assembly is loosely modeled on the metaphor of reversible self-assembly in the material world, In material self-assembly, local chaotic interactions between autonomous physical elements (biological cells, gas molecules, Wall street traders) produce global behavior that is well ordered. In process self-assembly, the atomic elements are virtual - autonomous, mobile fragments of code with state.

These process fragments (pfrags) use local messaging to emulate the forces that direct material self-assembly. Pfrags then use the-se simulated forces to direct their further migration, ultimately arriving at a predetermined spatial ordering. The shared memory model is patterned after bulletin board systems that communicate via lossy update channels.

Every particle contains a local entry to the bulletin board (the HomePage) where resident executables can read and write tagged data (posts). An additional segment of memory (the I/O space) is reserved for mirrored instances of the HomePages from the neighboring particles. Posts to a local HomePage appear - with an unbounded latency and non-zero probability of failure - at the

mirror sites on all the particles within the network neighborhood. For an executable running in a given particle, the HomePage is read-write and the I/O space is read-only.
All software executed in **paint** is represented as process fragments. Pfrags are self contained and sized to fit entirely in the RAM space of a single particle. Inter-pfrag messaging is coded as posts (variable length key-value pairs) and gated through the I/O. The pfrag's continual, free running evaluation of its environment (as encoded in the I/O space) is the key driver for adaptation on **paint.**

Ex.l. Answer the following questions:

1. How is computer graphics defined in the extract?
2. Why does computer graphics play a big role in information society technologies?
3. What is image-based rendering?
4. What are its applications?
5. How will you account for the acuteness of interactive visualization and display?
6. What is the personal everywhere visualizer?
7. What are its potential applications?
8. What are the limitations for real time display systems?
9. What is a paintable computer?
10. Explain the technical characteristics of the paintable computer and its programming mode.

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing:

Объёмная компьютерная графика играет всё большую роль в Интернете. С каждым годом всё больше и больше идёт сближение 3D- и Web-технологий. В результате выхода нескольких технологий "чистого" 3D-web, всплыли достаточно большие минусы такого подхода. К числу таких проблем относятся большие объёмы загрузки - Вы загружаете достаточно большие объёмы полигонов с текстурами. Также, обсчёт таких приложений производится практически одним процессором, поддержка стандартов ускорения 3D-графики минимальна или полностью отсутствует. Недаром, Intel рекламировал новые команды для работы с 3D-Web в процессорах Pentium III. Причём, если в мультимедиа-приложениях, таких как игры, подобные проблемы решают наращиванием мощностей, то скорости передачи каналов Интернета хоть и растут, но до сих пор недоступны для широкого круга пользователей.

Такую ситуацию на данный момент пытаются решить несколькими способами. Наиболее революционный метод - это переход на новые возможности 3D, которого ожидает вся индустрия компьютерной графики. А именно - замена технологии полигонов на технологию сплайнов и Nurbs-кривых.

Ex.4. Topics for discussion:

1. The future of computer graphics.
2. Innovations in computer graphics technologies.
3. The Information Society and its challenges for computer graphics development.

The Concepts of Programming Languages

Key vocabulary:

1. Floating-point computations - вычисления с плавающей точкой
2. Spreadsheet - (динамическая) электронная таблица, табличная программа
3. Embed - внедрять, встраивать
4. Type check - проверка типов, проверка соответствия типов
5. Assignment statement - оператор присваивания
6. Drag-and-drop - буксировка, перетаскивание
7. Declaration - объявление, описание
8. Control flow - управляющая логика, поток управления
9. Query processing - обработка запроса
10. Implementation - реализация, внедрение, разработка
11. Fetch-execute cycle - цикл выборки-выполнения
12. Source-level debugging operations - операции отладки, осуществляемые на уровне исходного кода
13. Imperative language - императивный язык, процедурный язык
14. Functional, or applicative language - язык функционального программирования, аппликативный язык.
15. Mark - up language - язык разметки

Computers have been applied to a myriad of different areas, from controlling nuclear power plants to providing video games in mobile phones. Because of this great diversity in computer use, programming languages with very different goals have been developed.

Scientific applications. The first digital computers, which appeared in the 1940s, were used and indeed invented for scientific applications. Typically, scientific applications have simple data structures but require large numbers of floating-point arithmetic computations. The most common data structures are arrays and matrices; the most common control structures are counting loops and selections. The early high-level programming languages invented for scientific applications were designed to provide for those needs. Their competition was assembly language, so efficiency was a primary concern. The first language for scientific applications was Fortran.

Business applications. The use of computers for business applications began in the 1950s. Special computers were developed for this purpose, along with special language. The first successful high-level language for business was COBOL, the initial version of which appeared in 1960. Business languages are characterized by facilities for producing elaborate reports, precise ways of describing and storing decimal numbers and character data, and the ability to specify arithmetic operations.

With the advent of personal computers came new ways for businesses to use computers. Two specific tools that can be used on small computers, spreadsheet systems and database systems, were developed for business and now are widely used, in both homes and businesses.

Artificial Intelligence. Artificial Intelligence (AI) is a broad area of computer applications characterized by the use of symbolic rather than numeric computations. Symbolic computation means that symbols, consisting of names rather than numbers, are manipulated. Also, symbolic computation is more conveniently done with linked lists of data rather than arrays. This kind of programming sometimes requires more flexibility than other programming domains.

The first widely used programming language developed for AI applications was the functional language LISP, which appeared in 1959. During the early 1970s, however, an alternative approach to some of these applications appeared - logic programming using

the Prolog language. More recently, some AI applications have been written in scientific languages such as C.
Systems programming. The operating system and all of the programming support tools of a computer system are collectively known as its systems software. Systems software is used continuously and therefore must be efficient. Therefore, a language for this domain must provide fast execution. Furthermore, it must have low-level features that allow the software interfaces to external devices to be written.
In the 1960s and 1970s, some computer manufacturers, such as IBM, Digital, and Burroughs (now UNYSYS), developed special machine-oriented high-level languages for systems software on their machines: PL/S, BLISS, and ALGOL.
The UNIX operating system is written almost entirely in C, which has made it relatively easy to port to different machines. Some of the characteristics of C make it a good choice for systems programming. It is low-level, execution-efficient, and does not burden the user with many safety restrictions.
Web **software.** The WWW is supported by an eclectic collection of languages, ranging from mark-up languages, such as XHTML, which is not a programming language, to general-purpose programming languages, such as Java. XHTML provides a way of embedding presentation instructions in the pages of information, which could include text, images, sound, or animation, that constitute Web content. These instructions are targeted to presentation devices, which could be browser displays, printers, or other devices. Because of the pervasive need for dynamic Web content, some computation capability is often included in the technology of content presentation. This functionality can be provided by embedding a programming code in an XHTML document. Such code is often in the form of a scripting language, such as PHP or Python.

Language Evaluation Criteria

In order to examine and evaluate the concepts of the various constructs and capabilities of programming languages it is necessary to have a set of evaluation criteria.

Readability. Perhaps one of the most important criteria for judging a programming language is the ease with which programs can be read and understood. Before 1970 the primary positive characteristics of programming languages were efficiency and machine readability. Language constructs were designed more from the point of view of the computer than of computer users. In the 1970s, however, the software life cycle concept was developed, maintenance was recognized as a major part of the cycle. Because ease of maintenance is determined by the readability of programs, readability became an important measure of the quality of programs and programming languages. There was a distinct crossover from a focus on machine orientation to a focus on human orientation.

Writability. Writability is a measure of how easily a language can be used to create programs for a chosen problem domain. It is simply not reasonable to compare the writability of two languages in the realm of a particular application when one was designed for that application and the other was not. For example, the writabilities of COBOL and Fortran are dramatically different for creating a program to deal with two-dimensional arrays for which Fortran is ideal. Their writabilities are also quite different for producing financial reports with complex formats, for which COBOL was designed.

Reliability. A program is said to be reliable if it performs to its specifications under all conditions. Type checking is an important factor in language reliability. The earlier errors in programs are detected, the less expensive it is to make the required repaires.

Cost. The ultimate total cost of a programming language is a function of many of its characteristics. First, there is the cost of training programmers to use the language. Second is the cost of

writing programs in the language. Both the cost of training programmers and the cost of writing programs in a language can be significantly reduced in a good programming environment. Third is the cost of compiling programs in the language. Fourth, the cost of executing programs written in a language is greatly influenced by that language's design. The fifth factor is the cost of the language implementation system. Sixth is the cost of poor reliability.
Portability. This is the ease with which programs can be moved from one implementation to another. Portability is most strongly influenced by the degree of standardization of the language, which is a time-consuming and difficult process.

Language Categories

The basic architecture of computers has had a profound effect on language design. Most of the popular languages of the past 50 years have been designed around the von Neumann architecture. These languages are called **imperative languages.**
Because of the von Neumann architecture, the central features of imperative languages are variables, assignment statements, and the iterative form of repetition. In these languages an algorithm is specified in great detail, and the specific order of execution of the instructions or statements must be included.
The syntax and the semantics of the imperative languages are very complex. In an imperative language, the programmer must make a static division of the program into its concurrent parts, which are then written as tasks. Concurrent execution in imperative language can be a complicated process. The most efficient imperative languages are C and Fortran.

A **functional, or applicative, language** is one in which the primary means of computation is applying functions to given parameters. Programming can be done in a functional language with-out the kind of variables that are used in imperative

languages, without assignment statements, and without iteration. This makes the syntax and the semantics of the functional languages simple compared to that of the imperative languages.
Programs in functional languages can be divided into concurrent parts dynamically, by the execution system, making the process highly adaptable to the hardware on which it is running. The closeness of functional programming to mathematics, while resulting in conciseness and elegance, may in fact make functional programming languages less accessible to many programmers.
The most prominent among these languages are: LISP, COMMON LISP, and Scheme, which is widely used to teach functional programming.

The latest step in the evolution of software development is object-oriented design. Object-oriented methodology begins with data abstraction, and adds inheritance and dynamic method binding. Inheritance greatly enhances the potential reuse of existing software, providing the possibility of significant increases in software development productivity. This is an important factor in the increase in the popularity of object-oriented languages, such as Smalltalk, Ada 95, Java, and C++.

Another kind of language, the visual language, forms subcategory of the imperative languages. The most popular visual language is Visual BASIC, which is now being replaced by Visual BASIC.NET. These languages include capabilities for drag-and-drop generation of code segments. The characterizing feature of a visual language provides a simple way to generate graphical user interfaces to programs.

Languages used for logic programming are called logic programming languages, or declarative languages, because prorams written in them consist of declarations rather. than assignments and control flow statements. These declarations are actually statements in symbolic logic.

Declarative semantics is considerably simpler than the semantics of the imperative languages.
Programming in a logic programming language is nonprocedural. Programs in such languages do not state exactly ***how*** a result is to be computed but rather describe the form of the result. The difference is that we assume the computer system can somehow determine ***how*** the result is to be computed. What is needed to provide this capability for logic programming languages is a concise means of supplying the computer with both the relevant information and a method of inference for computing desired result.
Logic programming in general and Prolog language in particular are a natural match to the needs of implementing an RDBMS: only a single language is required, the deductive capability is built in.
Prolog can be used to construct expert systems. It can easily fulfill the basic needs of expert systems, using resolution as the basis for query processing, using its ability to add facts and rules to provide the learning capability, and using its trace facility to in-form the user of the ‘reasoning’ behind a given result.

In recent years, a new category of languages has emerged, **a mark-up/programming hybrid languages.** Mark-up languages, including the most widely used mark-up language, XHTML, is not a programming language. It is used to specify the layout of information in Web documents.

Implementation Methods

Programming languages can be implemented by any of three general methods. At one extreme, programs can be translated into machine language, which can be executed directly on the computer. This method is called **a compiler implementation,** and has the advantage of very fast program execution, once the translation process is complete.

Pure interpretation lies at the opposite end (from compilation) of implementation methods. With this approach, programs are interpreted by another program called an interpreter, with no translation whatever. The interpreter program acts as a software simulation of a machine whose fetch-execute cycle deals with high-level language program statements rather than machine instructions.

Pure interpretation has the advantage of allowing easy implementation of many source-level debugging operations, because all run-time error messages can refer to source-level units. But the execution is 10 to 100 times slower than in compiled systems and it often requires more space.

Some language implementation systems are a compromise between compilers and pure interpreters, they translate high-level language programs to an intermediate language designed to allow easy interpretation. This method is faster than pure interpretation because the source language statements are decoded only once. Such implementations are called **hybrid implementation systems.**

Ex.l. Answer the following questions:

1. What application areas of programming languages can you speak of?
2. How can the choice of the programming languages for different fields be justified?
3. What are the criteria to evaluate the programming languages?
4. Give the short descriptions of the language categories.
5. What is the difference between the implementation methods?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

Из чего бы ни делались компьютеры через сто лет, можно не сомневаться, что они будут гораздо быстрее, чем сейчас. Если действие закона Мура сохранится, компьютеры станут в 74 квинтиллиона раз быстрее. Это сложновато себе представить. Впрочем, по всей вероятности, закон Мура окажется несостоятельным. Всё, что должно удваиваться каждые восемнадцать месяцев, рано или поздно наталкивается на какой-нибудь фундаментальный предел.

Но даже если компьютеры станут быстрее лишь в жалкий миллион раз, это приведёт к не менее радикальным подвижкам самых основ, на которых строятся языки программирования. Помимо всего прочего, появится больше применений для языков, которые сейчас считаются "медленными", то есть тех языков, которые не транслируются в очень эффективный код.

Несмотря на это, приложения, требующие высокой производительности, будут существовать всегда. Некоторые задачи, которые решаются с помощью компьютеров, порождаются самими компьютерами. Например, скорость, с которой необходимо обрабатывать видео, напрямую зависит от скорости, с которой машина способна его генерировать. Кроме того, существует класс задач, которые по определению обладают неограниченной способностью поглощать все доступные ресурсы: визуализация, криптография, моделирование.

Если некоторые приложения могут становиться всё менее эффективными, а другие по-прежнему будут пытаться "выжать из железа последнее", языкам предстоит отвечать за неуклонно расширяющийся спектр задач. И это уже начинает происходить. Существующие реализации некоторых популярных новых языков ошеломительно расточительны по меркам прошлых десятилетий.

Такое происходит не только с языками программирования. Это всеобщая историческая тенденция. Развитие техники даёт новым поколениям возможность делать вещи, которые раньше считались бы излишеством. Лет тридцать назад люди были бы потрясены, узнав, насколько обыденными станут в наше время междугородные телефонные звонки. А лет сто назад известие о том, что сейчас за один день посылка может преодолеть путь от Бостона до Нью-Йорка через Мемфис, поразило бы всех ещё сильнее.

Ex.4. Topics for discussion.

1. Programming languages categorization.
2. The diversity in programming languages application.
3. Language evaluation criteria and how they help to examine the programming languages capabilities.
4. Prepare the comparative analysis of two programming languages belonging to two different categories.

Software Security Basics

Key vocabulary:

1. Extensible system - расширяемая система
2. Vulnerability - уязвимость (системы)
3. Ubiquitous - распространенный, повсеместно встречающийся
4. Exacerbate (v.) - углублять, усиливать, обострять
5. Bogus - поддельный, фальшивый
6. Propagation - распространение, продвижение, передача
7. Intrusion detection system - система обнаружения вторжений
8. Authentication - проверка подлинности, проверка прав доступа, идентификация
9. Secure socket layer (SSL) - протокол безопасных соединений
10. Penetrate-and-patch approach - метод проникновения и исправления
11. Malicious - враждебный

The Internet continues to change the role that software plays in the business world, fundamentally and radically. Software no longer simply supports back offices and home entertainment. Instead, software has become the lifeblood of our businesses and has become deeply entwined in our lives. The invisible hand of Internet software enables e-business, automates supply chains, and provides instant, worldwide access to information. At the same time, Internet software is moving into our cars, our televisions, our home security systems, and even our toasters.

The biggest problem in computer security today is the software. Internet-enabled applications, including those developed internally by a business, present the largest category of security risk today. Real attackers compromise software. Of course, software does not need to be Internet enabled to be at risk. The Internet is just the most obvious avenue of attack in most systems.

Software is at the root of all common computer security problems. If your software misbehaves, a number of diverse sorts of problems can crop up: reliability, availability, safety, and security. Malicious hackers don't create security holes, they simply exploit them. Security holes and vulnerabilities - the real root cause of the problem - are the result of bad software design and implementation.

Technical Trends Affecting Software Security

One significant problem is the size and complexity of modern information systems and their corresponding programs. Complex systems, by their very nature, introduce multiple risks. And almost all systems that involve software are complex. Inherent complexity lets malicious subsystems remain invisible to unsuspecting users until it is too late. This is one of the root causes of the malicious code problem. For example, a desktop system running Windows/NT and associated applications depends on the proper functioning of the kernel as well as the applications to ensure that an attacker cannot corrupt the system. However, NT itself consists of approximately 35 million lines of code, and applications are becoming equally complex. When systems become this large, bugs cannot be avoided.
Exacerbating this problem is the widespread use of low-level programming languages, such as C or C++, that do not protect against simple kinds of attacks (most notably, buffer overflows). The complexity of a system makes it hard to understand, hard to analyze, and hard to secure. Security is difficult to get right even in simple systems; complex systems serve only to make security harder.
A second trend exacerbating software security problems is the degree to which systems have become extensible. From an economic standpoint, extensible systems are attractive because

they provide flexible interfaces that can be adapted through new components. In today's marketplace, it is crucial that software be deployed as rapidly as possible to gain market share. Yet the marketplace also demands that applications provide new features with each release. An extensible architecture makes it easy to satisfy both demands by letting companies ship the base application code early, and later ship feature extensions as needed.

Unfortunately, the very nature of extensible systems makes security harder. For one thing, it is hard to prevent a malicious code from slipping in as an unwanted extension. Furthermore, analyzing the security of an extensible system is much harder than analyzing a complete system that cannot be changed.

A **third trend** that has allowed software security vulnerabilities to flourish is the fact that computer networks are becoming ubiquitous. The growing connectivity of computers through the Internet has increased both the number of attack vectors and the ease with which an attack can be made. More and more computers, ranging from home PCs to systems that control critical infrastructures, are being connected to the Internet. People, businesses, and governments are increasingly dependent on network-enabled communication such as e-mail or Web pages provided by information systems. Unfortunately, because these systems are connected to the Internet, they become vulnerable to attacks from distant sources. An attacker no longer needs physical access to a system to cause security problems.

Because access through a network does not require human intervention, launching automated attacks from the comfort of your living room is relatively easy. Indeed, the well-publicized denial-of-service attacks in February 2000 took advantage of a number of hosts to flood popular e-commerce Web sites, including Ya-hoo!, with bogus requests automatically.

Together, the three trends of ubiquitous networking, growing system complexity, and built-in extensibility make the software security problem more urgent than ever.

Security Goals

Security is not a static feature on which everyone agrees. It is relative. Not only is there no such thing as 100% security, even figuring out what 'secure' means differs according to context. A key insight about security is to realize that any given system, no matter how 'secure', can probably be broken. In the end, security must be understood by thinking about **goals.** What is it we are trying to protect? From whom are we protecting it? How can we get what we want?

Prevention. Internet time compresses not only the software development life cycle, it also directly affects the propagation of attacks. Once a successful attack on a vulnerability is found, it spreads like wildlife on the Internet.

Internet time is the enemy of software security. Automated Internet-based attacks on software are a serious threat that must be factored into the risk management equation. This makes prevention more important than ever.

Auditing and monitoring. Because there is no such thing as 100% security, attacks will happen. One of the keys to recovering from an attack is to know who did what, and when they did it. Although auditing is not a direct prevention technology, knowing that there is a system for accountability may in some cases dissuade potential attackers.

Monitoring is real-time auditing. Intrusion detection systems can monitor programs for known signatures, such as dangerous patterns of low-level system calls that identify an attack in progress. More complex approaches place monitors in the code itself in the form of assertions.

Privacy and confidentiality. There are clear reasons for business, individuals and governments to keep secrets. The problem is, software is not really designed to do this. It is designed to run on a machine and accomplish some useful work. This means that a machine on which a program is running can pry out every secret a piece of software may be trying to hide.

One very simple, useful piece of advice is to avoid storing secrets like passwords in our code, especially if that code is likely to be mobile.

Multilevel security. Some kinds of information are more secret than others. Most governments have multiple levels of information classification, ranging from Unclassified and merely For Official Use Only, through Secret and Top Secret, all the way to Top Secret/Special Compartmentalized Intelligence.

Most corporations have data to protect too - sometimes from their own employees. Some business partners will be trusted more than others. Different levels of protection are afforded different levels of information.

Anonymity. Anonymity is a double-edge sword. Often there are good social reasons for some kind of anonymous speech (think of AIDS patients discussing their malady on the Internet), but just as often there are good social reasons not to allow anonymity (think of hate speech, terrorist threats, and so on).

Microsoft's Global Identifier tracks which particular copy of Microsoft Office originated a document. Sound like Big Brother? Consider that this identifier was used to help tie David L. Smith, the system administrator who created and released the Melissa virus, to his malicious code. What hurts us can help us too.

Authentication. Authentication is crucial to security because it is essential to know who to trust and who not to trust. It is a critical software security problem to take seriously.

People falsely believe that when the little lock icon on their browser lights up that they have 'a secure connection'. Secure socket layer (SSL) technology uses cryptography to protect the data stream between the browser and the server to which it is connected. But from an authentication standpoint, the real question to ponder is to *whom* are you connected? Clicking on the little key may reveal a surprise.

Integrity. When used in a security context, integrity refers to staying the same. By contrast to authentication, which is all about who, when and how, integrity is about whether something has been modified since its creation.

Digital information is particularly easy to fake. Sometimes it can be harder to print counterfeit money than to hack a stored-value smart card with differential power analysis (DPA) and to add some electronic blips. The more the new economy comes to rely on information, the more critical information integrity will become.
Good software security practices can help ensure that software behaves properly. We can avoid the Band-Aid-like penetrate-and-patch approach to security only by considering it as a crucial system property. This requires integrating software security into your entire software engineering process.

Ex.l. Answer the following questions:

1. Why does the Internet software present the largest computer security problem today?
2. What three trends make software security problem more urgent?
3. How can complex systems introduce multiple security risks?
4. Why is the Internet time the enemy of software security?
5. What are the goals of software security?
6. Why is anonymity referred to as “a double-edge sword”?

Ex.2. Give the summary of the text using the key terms.

Ex.3. Translate in writing.

Для чего кому-то нужно взламывать ваш компьютер?
Даже если вы самый что ни на есть обыкновенный пользователь, и на вашем компьютере нет какой-либо ценной и секретной информации, не нужно пребывать в иллюзии, что ваш компьютер никому не интересен. С точки зрения хакеров и людей, распространяющих вредоносные программы, он всё равно будет представлять ценность. Времена, когда вирусы писали ради спортивного интереса, уже прошли и сегодня весь хакерский инструментарий используется ради получения коммерческой выгоды. Сегодня вредоносные программы

стараются маскироваться и скрывать свою деятельность, чтобы втайне выполнять заложенные в них функции. Такими функциями могут быть: 1) кража паролей от ваших электронных кошельков, почтовых ящиков, icq, сайтов, аккаунтов в различных сервисах и т. д. 2) достаточно прибыльным "бизнесом" в наше время является организация DDoS-атак, которые могут направляться на любой сайт или сервер, даже не имеющий каких-либо существенных уязвимостей. В результате таких атак сервер перегружается запросами, идущими с многочисленных компьютеров в разных регионах мира и сайт, на который направлена атака, отключается. 3) организация массовых рекламных рассылок является, к сожалению, прибыльным бизнесом, и для таких целей также практикуется заражение компьютеров обычных пользователей троянами.

Источники опасностей. Подхватить вредоносную программу, к сожалению, значительно легче, чем многие себе представляют. Для взлома компьютеров пользователей сети и кражи важных данных, например, паролей электронных платёжных систем, применяются следующие методы:

1) социальная инженерия. Вам могут прислать письмо от имени администрации сервиса с просьбой выслать им якобы утерянный пароль или письмо, содержащее безобидный файл, в который на самом деле спрятан троян, в расчёте на то, что из любопытства вы сами его откроете и запустите вредоносную программу.

2) трояны и вирусы могут быть спрятаны в различных бесплатных, доступных для скачивания из Интернета, программах, которых огромное множество, или на пиратских дисках, имеющихся в свободной продаже.

3) взлом вашего компьютера может быть произведён через дыры в распространённом программном обеспечении, которых, к сожалению, довольно много, и всё новые уязвимости появляются регулярно. Хакеры, в отличие от большинства пользователей, не следящих за уязвимостями и часто не скачивающих устраняющие их патчи, за

обнаружением новых уязвимостей следят и используют их в своих целях.

Меры по защите.

1) Установите файрволл (firewall). 2) Установите антивирусное и антишпионское ПО. Антивирусное ПО должно запускаться автоматически при загрузке Windows и работать постоянно, проверяя запускаемые вами программы, в фоновом режиме. Обязательно проверяйте на вирусы перед первым запуском любые программы, которые вы где-либо скачиваете или покупаете. 3) Не устанавливайте или удалите лишние ненужные службы Windows, которые не используете. Это ограничит возможности хакеров по доступу к вашему компьютеру. 4) Не открывайте подозрительные письма странного происхождения, не поддавайтесь на содержащиеся в них сомнительные предложения лёгкого заработка, не высылайте никому пароли от ваших аккаунтов, не открывайте прикреплённые к письмам подозрительные файлы и не переходите по содержащимся в них подозрительным ссылкам.5) Не используйте простые пароли. Не используйте один и тот же пароль на все случаи жизни. 6) Будьте осторожны при выходе в Интернет из мест общего пользования (например, Интернет-кафе), а также при использовании прокси-серверов. Пароли, который вы вводите, в этом случае, с большей вероятностью могут быть украдены.7) При использовании электронных платёжных систем типа webmoney или Яндекс-деньги, работа с ними через веб-интерфейс является менее безопасной, чем если вы скачаете и установите специальную программу (webmoney keeper).

Ex.4. Topics for discussion.

1. The Internet is the most obvious avenue of attack.
2. Technical trends affecting software security.
3. The security goals.
4. Preventive measures are the key point in the provision of software security.

Appendix I

In the following tests choose the best possible answer (A), (B), (C) or (D) that best fits the given statements.

Database Basics Test

1 .This is a collection of data items organized as a set of formally-described tables from which data can be accessed or reassembled in many different ways without having to reorganize the database tables.
a. relational database
b. array
c. file allocation table
d. splay tree
2. This is a standard interactive and programming language for getting information from and updating a database.
a. Dynamic Data Exchange
b. Structured Query Language
c. ASCII
d. Erlang programming language
3. In creating a database, this is the process of organizing it into tables in such a way that the results of using the database are always unambiguous.
a. probability
b. normalization
c. data modeling
d. virtual organization

4. In 1977, this company set out to prove that the prevailing theory that relationship databases lacked commercial viability, was wrong.
a. Hewlett-Packard
b. Oracle

c. Cognos
d. Solaris
5. This family of products is IBM's cross-platform relational database management system.
a. DBA
b.PHP
c. RDF
d. DB2

6. This type of database, which can be used to mine data for business trends, doesn't require SQL queries, but instead allows a user to ask questions like "How many Aptivas have been sold in Nebraska so far this year?"
a. line information database
b. High Performance Storage System
c. object-oriented database management system
d. multidimensional database
7. This is an open standard application programming interface (API) for accessing a database.
a. Universal Data Access
b. Open Database Connectivity
c. Topic Map Query Language
d. Open Data-Link Interface
8. This term, used to describe a database that occupies magnetic storage in the terabyte range, describes a database that contains billions of table rows.
a. Very Large Database
b. holographic storage
c. Cold Fusion
d. giant
9. This is a collection of descriptions about data objects that is created for the benefit of programmers and others who might need to refer to them.
a. data dictionary
b. stored procedure

c. Virtual File Allocation Table
d. Virtual Address extension
10. This term is used to describe the process of forecasting, or simply discovering patterns in data that can lead to predictions about the future.
a. customer relationship management
b. PERT chart
c. data mining
d. enterprise risk management

Security Basics Test

1 .This technology is used to measure and analyze human body characteristics for authentication purposes.
a.footprinting
b.biometrics
c. anthropomorphism
d.optical character recognition
2.This is an electronic or paper log used to track computer activity.
a.traceroute
b.cookie
c. weblog
d.audit trail
3.This is a series of messages sent by someone attempting to break into a computer to learn which computer network services the computer provides.
a. bit robbing
b.Web services description language (WSDL)
c.port scan
d.service profile identifier
4.This is the name for a group of programmers who are hired to expose errors or security holes in new software or to find out why a computer network's security is being broken.
a. computer emergency response team

b. tigerteam
c. silicone cockroach
d. Defense Advanced Research Projects agency
5.This is a mechanism for ensuring that only authorized users can copy or use specific software applications.
a. authorized program analysis report
b. private key
c. access log
d. dongle
6. This is the hiding of a secret message within an ordinary message and the extraction of it at its destination.
a. secret key algorithm
b. message queueing
c. spyware
d. steganography
7. This is a Peripheral Component Interconnect card used to generate encryption keys for secure transactions on e-commerce.
a. smart card
b. server accelerator card
c. network interface card
d. intelligent peripheral interface
8. This is the name of a technology involving the monitoring of devices that emit electromagnetic radiation in a manner that can be used to reconstruct intelligible data.
a. reverse engineering
b. magnetoresistive head technology
c. van Eck phreaking
d. lurking
9. This enables users of a basically unsecure public network such as the Internet to securely and privately exchange data and money through the use of a public and private cryptographic key pair that is obtained and shared through a trusted authority.
a. Security Identifier
b. public key infrastructure
c. Internet Assigned Numbers Authority

d. Private Branch Exchange
10. This is an assault on the integrity of a security system in which the attacker substitutes a section of ciphertext (encrypted text) with a different section that looks like (but is not the same as) the one removed.
a. Trojan horse
b. hashing
c. swap file
d. cut and paste attack

Artificial Intelligence Test

1. This is a system of programs and data structures that approximates the operation of the human brain.
a. Intelligent Network
b. decision support system
c. neural network
d. genetic programming

2. This is the tendency for people to think of inanimate objects as having human-like characteristics.
a. aliasing
b. personalization
c. self-replication
d. anthropomorphism

3. This is a programming language that was designed for easy manipulation of data strings. It was developed in 1959 by John McCarthy and is still commonly used today in artificial intelligence (AI) programming.
a. LISP
b. assembly language
c. machine code
d. Ruby

4. This is an approach to computing developed by Dr. Lotfi Za-deh based on "degrees of truth" rather than the usual "true or false" (1 or 0) Boolean logic. Dr. Lotfi Zadeh developed this approach while working on the problems computers had under-standing natural language.
a. cyberwoozling
b. fuzzy logic
c. Smalltalk
d. arachnotaxis

5. This is a type of computer program that simulates the judgement and behavior of a human or organization that possesses expert knowledge and experience in a particular field.
a. expert system
b. cyborg
c. autonomous system
d. cybrarian

6. This is a program that allows the computer to recognize human movement such as waving, finger pointing, or change in eye direction and identify the motions as specific means of interaction.
a. MIME
b. show control
c. gesture recognition
d. motion plan

7. This is the ability of a computer to use binocular vision to differentiate between objects. The computer uses high-resolution cameras, a large amount of random access memory (RAM), and an artificial intelligence (AI) program to interpret data.
a. DiffServ
b. model-view-controller
c. machine vision
d. eye-in-hand system

8. This is a program that gathers information or performs some other service on a regular schedule without a human being's immediate presence.
a. aggregator
b. agile applet
c. page
d. intelligent agent

9. This is a program that allows the computer to simulate conversation with a human being. "Eliza" and "Parry" are early exam-ples of programs that can at least temporarily fool a real human being into thinking they are talking to another person.
a. Speech Application Program Interface
b. chatterbot
c. speech recognition
d. Amiga

10. This is the potential ability of the human brain to accept an implanted mechanical device, such as a computer, as a natural part of its representation of the body.
a. virtual machine
b. self-assembly
c. serendipity
d. brain-machine interface

Nanotechnology Test

1. Who is generally credited with the first serious scientific claim that manufacturing on the molecular or even the atomic scale was possible? The claim was made at California Technical Institute and was called, "There's Plenty of Room at the Bottom".
a. K. Eric Drexler
b. Ralph Merkle
c. Ed Regis
d. Richard P. Feynman

2. In 1986, Dr. K. Eric Drexler published a book for the layman that gave a wide overview of the potential applications of molecular nanotechnology in such areas as computing, medicine, space science, and the military. What was the name of this ground-breaking book?
a. The Atomic Cookbook
b. Smaller is Better
c. Engines of Creation
d. A Crowded Blueprint
3. A particular molecule of carbon made up of sixty carbon atoms has received some press as a structure that shows promise as a basic building block in the area of molecular manufacturing. What is the whimsical non-technical name for these molecules?
a. Buckyballs
b. Nanocubes
c. Nanonodes
d. Fullerrods
4. What is the general name for the class of structures made of rolled up carbon lattices?
a. Nanotubes
b. Nanosheets
c. Nanorods
d. Fullerrods
5. Nano, as a prefix, denotes what order of magnitude?
a. 10^-3
b. 10^-6
c. 10^-9
d. 10^-12
6. Wat is the term used in the field of nanotechnology to describe an as-yet theoretical device that "will be able to bond atoms together in virtually any stable pattern?"
a. Replicator
b. Assembler
c. Constructor
d. Stacker

7. In discussions of the potential of molecular nanotechnology, the possibility has been posited that badly or maliciously designed self-assembling structures could get out of control, and destroy or disassemble all structures they encounter in their blind quest to replicate. What is the term for such a structure or group of structures?
a. Blue goo
b. Gray goo
c. Red goo
d. Green goo
8. Scientists discussing the potential of molecular nanotechnology realized the possibility that self-assembling molecular constructs could conceivably get out of control and destroy just about anything. This led to the concept that other constructs could be designed to neutralize and/or destroy the rogue substances before they got out of hand. By what colorful term are these theoretical "antibody" substances collectively known?
a. Gray goo
b. Blue goo
c. Red goo
d. Green goo
9. Many challenges exist to be overcome before molecular manufacturing can truly reach maturity as an applied science. Which of the following is such a challenge when designing molecular machinery?
a. Thermal noise
b. Complexity of design
c. Quantum fluctuation
d. All of these
10. As of public record at the end of 2002, which country was making the greatest annual investment in molecular nanotechnology research?
a. United States
b. Japan
c. South Korea
d. Russia

Appendix II
Как писать научный реферат по-английски

Реферирование

Реферат (от лат. refero - сообщаю) - это краткое изложение в письменном виде или в форме публичного доклада содержания научного труда и литературы по теме. В реферате излагается содержание печатного труда с характеристикой методов исследования, с фактическими данными и итогами работы. Реферирование предполагает владение мастерством сокращения текста первичного материала.

Реферат включает библиографическое описание первичного материала, саму часть (текст реферата), примечания референта, его собственное мнение относительно обозреваемых вопросов.

Приступая к работе над рефератом, сначала следует прочитать используемые материалы, с целью понять их основную идею или основные положения (Scheming reading). После повторного, более внимательного аналитического чтения того же самого текста (Close reading), следует начать с составления плана, выделив основную идею работы в целом и каждого абзаца в отдельности. Далее, в каждом абзаце (Logical unit) находим одно или два предложения, которые подтверждают основную мысль каждой логической единицы.

Изучив и проработав используемые материалы, Вы приступаете к работе над своим рефератом. Мы рекомендуем построить его по следующему плану:

1. Цель и предмет реферата;
2. Методы работы;
3. Характеристика и составление используемых материалов;
4. Ваши собственные выводы.

Цель и предмет реферата

Для обозначения цели и предмета реферата можно употребить следующие существительные: **the aim, the object, the purpose, the task** - цель, назначение, задача (употребляются с определенным артиклем, поскольку мы говорим не о целях вообще, а о цели данной работы).

Прилагательные **main, chief, primary, principal** (главный, основной) могут определять существительные **aim, purpose, etc.**

Если Вам надо сообщить о цели своей работы, Вы можете использовать следующую конструкцию:

Purpose, aim, etc. + be + The Infinitive

Пример:

The aim of the study is to determine...	Цель работы состоит в определении.../Целью работы является определение...

При сообщении о предмете исследования Вам понадобятся следующие глаголы:

Study - изучать, исследовать
Investigate - тщательно, всесторонне исследовать, расследовать
Examine - рассматривать, проверять, изучать
Analyze - изучать, анализировать
Consider - изучать, рассматривать
Describe - описывать
Discuss - обсуждать, излагать, полемизировать
Outline - кратко описывать, очерчивать
Obtain - получать
Determine - определять, находить
Find - находить, обнаруживать
Establish - устанавливать, точно определять.

При составлении реферата Вам могут понадобиться следующие сочетания:

to undertake/ to **perform a study** - исследовать, изучать

to carry out an investigation - проводить исследование
to perform the analysis of - проводить анализ
to give the description of - давать описание
to pay attention to - обращать внимание на...
to emphasize/to give emphasis to/ to place the emphasis on - подчеркивать.
Значение сочетаний можно усилить следующими прилагательными и наречиями: **particular, special, specific особый; great -** большой; **primary -** первостепенный; **particularly, specially, specifically.**

Методы работы

Для характеристики метода (методики) работы можно использовать следующие существительные:
Method - метод, способ
Technique - методика, техника, способ
Techniques - методика
Procedure - прием, процедура, операция
Approach - подход, рассмотрение
Differential method - дифференциальный метод
Isotopic technique - изотопный метод
Trial-and-error procedure - метод проб и ошибок
Device - небольшой прибор
Instrument - измерительный прибор
Apparatus - аппаратура, аппарат
Equipment - оборудование
Set - установка, агрегат
Unit - узел установки, агрегат
Construction - сооружение
Design - конструкция
Tool - орудие, приспособление
Facility - предметы оборудования, устройство.
Вам, возможно, будут полезны и следующие прилагательные:
Direct, straightforward - прямой

Indirect - косвенный
Elaborate - тщательно разработанный
Effective - эффективный, результативный
Efficient - экономичный
Versatile - разносторонний
Valid - обоснованный, имеющий силу
Conventional - обычный, общепринятый
Unconventional - нестандартный
Convenient - удобный
Sensitive - чувствительный
Appropriate - подходящий, соответствующий
Если перед существительными **method, theory, effect, device** стоит имя собственное в родительном падеже, то артикль не употребляется: **Seitz's hypothesis, Hilbert's method.**
Если имя собственное стоит в общем падеже, то употребляется определенный артикль: **the Hall effect, the Boltzman factor.**
При оценке работы, метода, материала Вам понадобятся существительные:
Advantage, merit - достоинство, преимущество
Limitation, disadvantage, drawback - недостаток, недочет, ограничение.

Характеристика и сопоставление используемых материалов

Характеризуя и оценивая работу, мы можем указывать на погрешности и ошибки, при этом используя следующие слова и словосочетания:
Error - ошибка, погрешность
The source of the error - источник ошибки
To be in error by some factor (by a factor of two) - быть неверным во сколько-то раз (и два раза)
Errors are due to - ошибки вызываются

Errors arise due to - ошибки возникают из-за
The error effects - погрешность влияет на
An error is introduced - ошибка вкрадывается
Reduce an error - уменьшать погрешность
Eliminate an error - устранять погрешность
Основными глаголами для выражения сопоставления являются: **compare with, make/give a comparison of...with -** сравнивать, проводить сравнение с..., **make/give a comparison between...and...** - проводить сравнение между...и...
При сопоставлении чего-то нового, современного со старым Вам могут понадобиться следующие слова и выражения:
At present - в настоящее время, теперь
Recently - недавно
Former - прежний
Earlier, formerly, previously - ранее, прежде
A method generally (universally) used, employed, adopted - общеупотребительный, общепринятый метод
Inferred from - выведенный из, полученный из
Differ from ...in/by/in that ... - отличаться от... по какому-либо признаку, свойству; на величину, числовое значение; отличаться тем, что...
При сопоставлении мнения автора используемых материалов, с одной стороны, и мнения автора реферата, с другой стороны, могут быть также использованы следующий выражения: **it is well known that, as is known, according to, in his opinion, to my mind, on the contrary, in turn, in the sense that, likewise, in short, at the angle of, from his point of view, etc.**

Ваши собственные выводы

Переходя к заключительной части реферата с Вашими выводами, можно использовать следующие выражения:
Make, draw, reach a conclusion, come to a conclusion - делать заключение (вывод) относительно...

From the results it is concluded that - на основании полученных результатов приходим к выводу, что ...
Lead to a conclusion, make it possible to conclude that, Concerning, as to... - приходить к заключению, давать возможность заключить, что...
It may be noted that - можно отметить, что
It may be stated that - можно утверждать, что
Thus, therefore, consequently, as a result - таким образом, следовательно, в результате

В заключение хотим дать еще некоторые рекомендации:
1. Как известно, для письменной научной прозы характерны сложные синтаксические конструкции с причастными, герундиальными и инфинитивными оборотами. Задача заключается в том, чтобы преобразовать сложные предложения в более простые, а последние, в свою очередь, сократить до “ядра предложения”, то есть, до слов, несущих основную смысловую нагрузку **(kernel sentences).** Для того, чтобы сохранить логическую структуру оригинала, необходимо соединить предложения **(kernel sentences)** в логически построенный ряд с помощью связующих слов **(connective words).** К таким словам можно отнести такие, как: **then, therefore, thus, because, yet, moreover, as well as, also, in the case of, consequently, hence, in order to, since, that is why, nevertheless, however, besides, in addition to, etc.**
2. Реферат является Вашей первой научной работой. В конце реферата необходимо представить список используемых материалов **(References),** включающий:
а) название работы;
б) год и место ее издания.

The List of Acronyms and Abbreviations

1. AI (Artificial Intelligence) - искусственный интеллект
2. ALGOL (Algorithmic Language) - алгоритмический язык
3. API (Application Programming Interface) - прикладной программный интерфейс
4. AR (Automatic Restoration) - автоматическое восстановление
5. ATM (Automatic Teller Machine) - банкомат
6. BASIC (Beginner's All Purpose, Symbolic Instruction Code) - универсальный символический язык для начинающих программистов
7. COBOL (Common Business Oriented Language) алгоритмический язык для экономических и коммерческих задач
8. CPU (Central Processing Unit) - центральный процессор
9. CRT (Cathode-Ray Tube) - электронно-лучевая трубка
10. DBA (Database Administrator) - администратор базы данных
11. DBMS (Database Management System) - система управления базами данных
12. DSS (Decision Support System) - система поддержки принятия решений
13. EDP (Electronic Data Processing) - электронная обработка данных
14. FET (Frontier Enabling Technologies) - технологии, открывающие новые области
15. Fortran (Formula Translation) - алгоритмический язык для научных задач
16. HCI (Human-Computer Interface) - интерфейс "человек-машина"
17. IBR (Image-Based Rendering) - рендеринг, визуализация, основанная на анализе изображения

18. ICT (Information and Communication Technologies) - информационные и телекоммуникационные технологии
19. ID (Identificator) - идентификатор
20. I/O (Input-Output) - ввод-вывод
21. IQ (Intelligence Quotient) - коэффициент интеллекта
22. IS/IT (Information system/information Technology) - информационная система/информационная технология
23.IST (Information Society Technologies) - технологии информационного общества
24. MIS (Management Information System) - административная информационная система
25. MNT (Molecular Nanotechnology) - нанотехнология на молекулярном уровне
26. MRI (Magnetic Resonance Imaging) - получение изображений органов тела с помощью магнитного резонанса
27. NASA (National Aeronautics and Space Administration) - Национальное управление по аэронавтике и исследованию космического пространства, НАСА (США)
28. OO (Object-Oriented) - объектно-ориентированный
29. PC (Personal Computer) - персональный компьютер
30. PET (Positron Emission Tomography) - позитронно-эмиссионная томография
31.PHP (Personal Hypertext Preprocessor) - персональный гипертекстовый обработчик
32. RAM (Random Access Memory) - оперативная память
33. RDBMS (Relational Database Management System) - реляционная система управления базами данных
34. SPM (Scanning Probe Electron Microscope) - **сканирующий** электронный микроскоп
35. SQL (Structured Query Language) - язык структурированных запросов
36. SSL (Secure Socket Layer) - протокол безопасных соединений
37. WWW (World Wide Web) - “Всемирная паутина”, глобальная гипертекстовая система Интернета

Bibliography

1. Mullins S. Craig. Database Administration. The Complete Guide to Practices and Procedures. - Boston: Addison-Wesley, **2002**.

2. Riccardi Greg. Principles of Database Systems with Internet and Java Applications. - Boston: Addison-Wesley, 2001.

3. Sebesta W. Robert. The Concepts of Programming Languages. - Boston: Addison-Wesley, 2006.

4. Summer Mary. Computers. Concepts and Uses. - New Jersey, 1988.

5. Viega John, McGraw Gary. Building Secure Software. How to Avoid Security Problems the Right Way. - Boston: Addison-Wesley, 2002.

6. Walker Tricia. Computer Science. - Oxford Polytechnic, England, 1989.

7. Whitworth Brian. Some Implications of Comparing Brain and Computer Processing. - Massey University, Auckland, New Zealand. - Proceedings of the 41st Hawaii Conference on System Sciences, 2008.

8. Journal of the American Society for Information Science and Technology. - Volume 59, Issue 9, July 2008.

9. Kokurina Elena. An Active Mind is the Key to Longevity. - The Moscow News, март-апрель, 2006.

10. Periscope Review - World news. Учебное пособие по английскому языку, выпуск № 19. - Волгоград, “Ритм планеты”, 2007

11. Гольцова Е.В. Английский язык для пользователей ПК и программистов. Самоучитель- Санкт-Петербург, “Корона-Век”, 2007.

12. Грэм Пол. Языки программирования через сто лет. - Компьютерра Online - 3 августа 2004 г.

13. Ежов А.А., Шумский С.А. Нейрокомпьютинг и его применения в экономике и бизнесе - Интернет Университет Информационных Технологий, 2007.

14. Магаршак Юрий. Избавление от бананотехнологий. - “Независимая газета”, 12.03.2008.

15. Ричлак Дж. Искусственный интеллект и человеческий разум. - Науковедение, выпуск № 4, - Москва, 1997.

16. Успенская Н.В., Михельсон Т.Н. “Как писать по-английски научные статьи, рецензии и рефераты”. - Санкт-Петербург, 1995.

Учебно-методическое издание

ENGLISH READER IN COMPUTER SCIENCE

СБОРНИК СТАТЕЙ ПО ИТ НА АНГЛИЙСКОМ ЯЗЫКЕ

Авторы-составители:
Кашелкина О.А., Круглова М.А.

Под редакцией *Кругловой М.А. и Саратовской Л.Б.*

Учебно-методическое пособие
для студентов второго курса

Издательский отдел
Факультета вычислительной математики и кибернетики
МГУ имени М.В. Ломоносова
Лицензия ИДК 05899 от 24.09.01 г.

119992, ГСП-2, Москва, Ленинские горы,
МГУ имени М.В. Ломоносова, 2-й учебный корпус

Напечатано с готового оригинал-макета
Издательство ООО “МАКС Пресс”
Лицензия ИД N 00510 от 01.12.99 г.
Подписано к печати 23.09.2008 г.
Печать офсетная. Бумага офсетная.
Формат 60x90 1/16. Усл.печ.л. 8,0. Тираж 650 экз. Заказ 513.

119992, ГСП-2, Москва, Ленинские горы, МГУ им. М.В. Ломоносова,
2-й учебный корпус, 627 к.
Тел. 939-3890, 939-3891. Тел./Факс 939-3891.