Н.В. Рибко, Н.А. Насонова

ТЕРМІНОЗНАВСТВО

ЧАСТИНА 2

Міністерство освіти і науки України
Вінницький національний технічний університет

Н.В. Рибко, Н.А. Насонова

ТЕРМІНОЗНАВСТВО

ЧАСТИНА 2

Затверджено Вченою радою Вінницького національного технічного університету як навчальний посібник для студентів напрямів підготовки 0804, 0915, 0160 з поглибленим вивченням іноземної мови. Протокол N 5 від 24 листопада 2005 р.

Вінниця ВНТУ 2006

УДК 811.111:04
Р 93

Рецензенти:

І. В. Степанова, кандидат філологічних наук, доцент ВНТУ
В. І. Месюра, кандидат технічних наук, доцент ВНТУ
С. В. Гладьо, кандидат філологічних наук доцент ВДПУ

Рекомендовано до видання Вченою радою Вінницького національного технічного університету Міністерства освіти і науки України

Рибко Н.В., Насонова Н.А.
Р 93 **Термінознавство.** Навчальний посібник для студентів напрямів підготовки 0804, 0915, 0160 з поглибленим вивченням іноземної мови. Частина 2. - Вінниця: ВНТУ, 2006. - 101с.

Навчальний посібник знайомить студентів зі спеціалізованими термінами з комп'ютерної техніки, що дає змогу розуміти фахову літературу та спілкуватися на теми за фахом.

В посібнику використовуються різноманітні вправи роботи з лексикою. Підбір завдань відкриває широкі можливості як для використання посібника на заняттях, так і для самостійної роботи студентів.

Навчальний посібник призначений для студентів, які вивчають комп'ютери, комп'ютерні технології, програмування, та широкого кола бажаючих оволодіти комп'ютерною лексикою.

УДК 811.111:04

CONTENTS

INTRODUCTION

This textbook is designed as an ESP course (English for Specific Purposes) for students whose future specialty is connected with computer sciences, computer engineering, computer technologies, programming, etc., and who will need English for occupational purposes.

The purpose of this textbook is to provide the students with knowledge of computer terms and practice of speaking and discussing topical issues in their field of specialization. There is a great social demand for specialists competent in their field of training and fully proficient in the language they use. This course emphasizes the vocabulary connected with computers. It consists of fourteen chapters, which are divided into two parts. It does not just contain language exercises; it is a kind of attempt to combine language learning and computer studying. Designed for students studying computer sciences, computer engineering, computer technologies, programming, this textbook is far from regarding the aim of training specialists in English in the field of their specialization. It is intended for those who have already acquired the fundamental knowledge connected with computers and computer technologies. But it implies further development of students' knowledge in their field of study, discussing urgent issues of the computer world and computer technologies.

Every chapter consists of three parts. The PRE-READING part makes the students recollect everything connected with the topic of the chapter in their native language what helps them both to focus on the needed technical terms, to study them, and to focus on the information presented in the texts. The next part READING contains texts and sentences for translation from English into Ukrainian, and from Ukrainian into English, a lot of exercises helping memorize terms, and preparing the students for discussion. The activities and exercises provide with extended practice on a given topic. The chapters contain many activities for studying specialized terms to instill the desire to speak, to express opinions. DISCUSSING is a part of the chapter that sums up what the students learnt in the chapter. It makes the students discuss the issues, encourage the students to develop their knowledge using Internet and other sources of information. Although terms and expressions play an important role in providing the students with tools they need, they should be viewed only as tools subordinate to the major purpose – the largest block of time is devoted to allowing the students freedom to express themselves in meaningful practice.

The pace of development of computer technologies is very fast, that's why there is the opportunity in one of the tasks to reconsider the information presented in the chapter and to discuss the problems with the group mates. A lot of texts, exercises, the glossary and dictionaries of computer terms turn this course into informative one, as well as practical one for students, teachers and specialists dealing with computer technologies, computer engineering, programming and dealing with computers in general.

UNIT 1

MICROCOMPUTERS

I PRE-READING

1. Recollect everything you studied concerning the topic, make notes. Write down the list of words in the Ukrainian language you think you will need while speaking on the topic.

2. a). Read the text and try to guess whether there are words from your list in it.

b). If you don't know the words, translate them using the dictionaries.

c). Choose the words from the list which are the most relevant for this topic.

d). If you don't know the meaning of any words or abbreviations, find their explanations in the Glossary.

e). Which words from exercises 1, 2c may be considered as general notions and which ones as technical terms?

f). Give explanations of several terms, using the Glossary.

II READING

1. Read and translate the text into Ukrainian.

Microcomputers, the smallest and least expensive computers, differ from other types of computers in capability, price, and size. The distinction between microcomputers and minicomputers fades as today's microcomputers become more powerful.

The increased power and miniaturization of microprocessors paved the way for the development of the microcomputer. The first microprocessors could manipulate 4 bits of data at a time. Most microcomputers today can manipulate 8 or 16 bits. Researchers have recently developed 32-bit microprocessors.

MITS developed the Altair 8800, the first microcomputer available for commercial sale. Pioneers influential in making microcomputers available to the general public include Ed Roberts of MITS, Stephen Wozniak and Steven Jobs who built the Apple II microcomputer, Jack Tramiel of Commodore Business Machines, John Roach of Tandy Corporation and John Opel, chief executive officer of IBM.

The speed of a microcomputer depends on two major factors: word size and clock speed. Word size refers to the number of bits that a computer can manipulate

at one time. Clock speed refers to the number of electronic pulses the microprocessor can produce each second.

Popular operating systems include CP/M, MS-DOS, Apple DOS and Apple ProDos, and Unix. No standard operating system for microcomputers exists. A coprocessor can make two microcomputers with different operating systems compatible. A coprocessor works with the computer's microprocessor to allow both operating systems to run on the microcomputer.

Common microcomputer input and output devices include monitors, joysticks and game paddles, the mouse, graphics tablets, touch screens, printers, and plotters. Common storage media include cartridge tapes, cassette tapes, and floppy disks. Large data storage requirements call for the use of hard disks.

Lightweight portable computers, briefcase, notebook, or hand-held size, do not require an external power source. Transportables are generally larger than portable computers and require an external power source, but are light enough to be carried.

Supermicrocomputers rival minicomputers in power but cost less. Supermicrocomputers provide users with high performance at a relatively low cost.

Microcomputers affect our lives in many ways. They have many uses in business, at home, and in schools. The capability of linking microcomputers by communication lines has triggered the development of distributed processing.

Commercial data bases provide microcomputer users with many types of information such as daily news, stock market reports, reviews and encyclopedias. Computer enthusiasts, also known as sysops, operate electronic bulletin boards that enable users of a particular system to share messages, information, and programs. Local area networks can link microcomputers in the same general area by a cable hookup. Some people predict that such telecommunications will create a "paperless" society and affect interpersonal relationships.

Owners of the same microcomputer model sometimes form a user's group to meet informally and exchange hardware, software, service, and support information. Computer stores meet the needs of small business and personal computer owners. A good computer store offers a variety of hardware and software and trained salespeople to answer questions for the computer novice.

Preventive maintenance keeps a microcomputer in good working order. Simple cleaning techniques for the monitor, keyboard, disk drive, and printer prevent serious problems.

2. Make up the logical scheme of the text and render the content of the text based on the scheme.

3. Fill in the gaps with the words from the list.

a. telecommunications	f. portables
b. operating system	g. microcomputer
c. transportables	h. clock speed
d. commercial data base	i. UNIX
e. MS-DOS	j. user's group

1. _____ are very powerful for their size.
2. The _____tells the speed at which program instructions are executed.
3. A(n) _____provides an interface between computer user's and computer hardware.
4. _____is not compatible with many microprocessors and is difficult for beginners to use.
5. _____can be adapted to many computers and is preferred by many programmers.
6. _____ come in briefcase, notebook, and hand-held size.
7. _____ are light enough to carry, yet can do almost as much as some desktop microcomputers.
8. _____ is the electronic transmission of data from one location to another.
9. People doing in-depth research often use _____.
10. The value of _____ comes from the accumulation of knowledge and experience of members.

4. Translate the sentences into the Ukrainian language. Take into consideration the information presented in the sentences. It may be helpful to you while doing the next exercises and discussing the topic.

1. The microprocessor is a single chip which also controls the storage of data, instructions, and the intermediate and final results of processing, much as the CPU of a main frame does. 2. The market for microcomputers continues to grow, although at a slower rate. 3. The microprocessor can manipulate a certain number of bytes of data without switching from primary storage to a supplementary storage bank. 4. A coprocessor is a microprocessor that can be plugged into the original computer to replace or work with the original microprocessor. 5. Local area networks, commercial data bases, and electronic bulletin boards are all associated with the process of telecommunication. 6. In 1974 MITS introduced the first microcomputer available to the public. 7. Word size is the number of bits that can be manipulated at one time. 8. Unix, first implemented on a microcomputer in 1978, may become the industry standard. 9. Portables do not need an external source of power, but usually need some form of direct-access

mass storage medium, such as floppy disks. 10. The electronic pulses built into the microprocessor affect the speed that instructions are executed because the instructions are executed at set intervals, which are timed by the electronic pulses.

5. Choose the right answer.

1. Microcomputers are approaching minicomputers in_____.
   a. capability
   b. price
   c. size
   d. home use

2. A microprocessor, which controls the sequence of operations and the arithmetic and logic operations of a microcomputer, can be compared to the _____.
   a. microcomputers of the 1970s
   b. a calculator
   c. the CPU of a mainframe computer
   d. minicomputer

3. The first microprocessor could manipulate 4 bits of data at a time. A recently developed microprocessor can manipulate _____bits of data at a time.
   a. 8
   b. 16
   c. 32
   d. 64

4. The first microcomputer available to the general public, the Altair 8800, had _____K of memory.
   a. 1
   b. 16
   c. 24
   d. 32

5. For the next several years, the price of microcomputers should continue to decline while _____
   a. their portability decreases
   b. more capabilities are added
   c. more businesses purchase microcomputers
   d. the demand for home computers decreases

6. A 16-bit microprocessor can manipulate twice as much data as an 8-bit microprocessor in approximately _____.
   a. half the time
   b. the same amount of time
   c. twice the time
   d. half the time and twice the accuracy

7. Microprocessor clock speed, measured in megahertz, is the number of electronic pulses the chip can produce each_____.
   a. tenth of a second
   b. second
   c. ten seconds
   d. minute

8. Supplementary storage banks should not be confused with secondary storage devices. Supplementary storage banks are part of_____.

a. the central processing unit
b. random-access memory
c. a microprocessor chip
d. primary storage

9. An operating system is a collection of programs used by the computer to _____.
a. perform operations in proper sequence
b. provide an input language and perform operations
c. manage its operations and provide an interface between user and hardware
d. manage its operations and translate calculations into user readable form

10. What operating system, not designed for any specific microprocessor, can be adapted to a new computer and allows a user to do several tasks at one time from the same terminal?
a. Unix, developed by Bell Laboratories
b. CP/M, developed by Digital Research
c. MS-DOS, developed by Microsoft
d. Apple DOS, developed by Apple

11. In order to use software or peripherals not compatible with a particular microcomputer, a user must first add a(n) _____.
a. machine language translator
b. replacement operating system
c. compatibility device
d. coprocessor

12. Computers that are light enough to carry, but require an external power source are called _____computers.
a. portable
b. movable
c. home computers
d. transportable

13. One problem hindering the full-scale implementation of supermicrocomputers is_____.
a. the expense of building dedicated microprocessors
b. the limited amount of available software
c. the difficulty in linking them to minicomputers
d. the expense of CPU's for supermicrocomputers

14. Which of the following is not associated with telecommunications?
a. local area networks
b. commercial data bases
c. electronic bulletin boards
d. storage media

15. Which of the following statements is false?
a. Maintenance experts emphasize that maintenance does not include do-it-yourself repair.
b. Of all computer components, keyboards have the greatest potential for problems.
c. A computer is only as reliable as the storage media used to store data.

d. Even in an ideal environment, dust particles and stable electricity are difficult to keep out of the computer.

6. Translate the words and word-combinations into English.

Касетна стрічка, співпроцесор, джерело живлення, монітор, портативний комп'ютер, дистанційний зв'язок, група користувачів, сумісність, твердий диск, носій пам'яті, джойстик, сенсорний екран, графічний планшет, тактовий імпульс, касетна стрічка, тактова частота, системний оператор, мишка, однокоординатний маніпулятор, мегагерц.

7. Say true or false and explain.

1. T F The prefix micro should be thought of as applying more to size and cost than to capability.

2. T F The speed with which the microprocessor can execute instructions affects the size of the microcomputer.
3. T F The first computers had large and cumbersome operating systems.
4. T F A coprocessor can make an operating system compatible with different peripherals.
5. T F In business, the use of microcomputers is limited primarily to executives using them for decision making.

6. T F Commercial data bases often use passwords to ensure that only paying customers can access a commercial data base.

7. T F Anyone with a microcomputer, a modem, communications software, and a telephone, can start an electronic bulletin board.

8. T F The membership dues of user's groups prohibits many computer owners from joining them.
9. T F Floppy disks are very sensitive to extreme heat and cold.
10. T F Floppy disk drives are durable and inexpensive to replace.

III DISCUSSING

1. Give the short answer.

1. Explain the technology that paved the way for microcomputers as they exist today.
2. List four companies involved in pioneering microcomputers.
3. What does a megahertz measure?
4. How did users of the first computers enter data and instructions?
5. Define compatibility in terms of computers.
6. List five input devices and two output devices.

7. Compare the advantages of cassette tapes and floppy disks as storage mediums.
8. What distinguishes standard microcomputers from the portables and transportables?
9. List three data communication systems made possible by telecommunications.
10. What basic preventive maintenance measures should microcomputer owners keep in mind?

2. Prepare reports using Internet or other sources (catalogues, magazines, books, etc.) about the latest news, achievements in the field concerning the topic of the chapter.

3. Summarize everything you have learnt on the topic.

4. Answer the question, taking into consideration quick pace of the latest achievements and developments in computer engineering and programming and information presented in the reports what information from the texts of the Chapter should be reconsidered, changed, added, etc…

5. It's interesting to know…

How Microprocessor Works

The computer you are using to read this page uses a microprocessor to do its work. The microprocessor is the heart of any normal computer, whether it is a desktop machine, a server or a laptop. The microprocessor you are using might be a Pentium, a K6, a PowerPC, a Sparc or any of the many other brands and types of microprocessors, but they all do approximately the same thing in approximately the same way.

If you have ever wondered what the microprocessor in your computer is doing, or if you have ever wondered about the differences between types of microprocessors, then read on. In this article, you will learn how fairly simple digital logic techniques allow a computer to do its job, whether it's playing a game or spell checking a document!

Microprocessor History

A microprocessor -also known as a CPU or central processing unit - is a complete computation engine that is fabricated on a single chip. The first microprocessor was the Intel 4004, introduced in 1971. The 4004 was not very powerful - all it could do was add and subtract, and it could only do that 4 bits at a time. However, it was amazing that everything was on one chip. Prior to the 4004, engineers built computers either from collections of chips or from discrete components (transistors wired one at a time). The 4004 powered one of the first portable electronic calculators.

The first microprocessor to make it into a home computer was the Intel 8080, a complete 8-bit computer on one chip, introduced in 1974. The first microprocessor to make a real splash in the market was the Intel 8088, introduced in 1979 and incorporated into the IBM PC (which first appeared around 1982). If you are familiar with the PC market and its history, you know that the PC market moved from the 8088 to the 80286 to the 80386 to the 80486 to the Pentium to the Pentium II to the Pentium III to the Pentium 4. All of these microprocessors are made by Intel and all of them are improvements on the basic design of the 8088. The Pentium 4 can execute any piece of code that ran on the original 8088, but it does it about 5,000 times faster!

Inside a Microprocessor

To understand how a microprocessor works, it is helpful to look inside and learn about the logic used to create one. In the process, you can also learn about assembly language - the native language of a microprocessor - and many of the things that engineers can do to boost the speed of a processor.

A microprocessor executes a collection of machine instructions that tell the processor what to do. Based on the instructions, a microprocessor does three basic things:

- Using its ALU (Arithmetic/Logic Unit), a microprocessor can perform mathematical operations like addition, subtraction, multiplication and division. Modern microprocessors contain complete floating-point processors that can perform extremely sophisticated operations on large floating-point numbers.
- A microprocessor can move data from one memory location to another.
- A microprocessor can make decisions and jump to a new set of instructions based on those decisions.

This microprocessor has:

- An address bus (that may be 8, 16 or 32 bits wide) that sends an address to memory
- A data bus (that may be 8, 16 or 32 bits wide) that can send data to memory or receive data from memory
- An RD (read) and WR (write) line to tell the memory whether it wants to set or get the addressed location
- A clock line that lets a clock pulse sequence the processor
- A reset line that resets the program counter to zero (or whatever) and restarts execution

Let's assume that both the address and data buses are 8 bits wide in this example.

Here are the components of this simple microprocessor:

- Registers A, B and C are simply latches made out of flip-flops. The address latch is just like registers A, B and C.
- The program counter is a latch with the extra ability to increment by 1 when told to do so, and also to reset to zero when told to do so.
- The ALU could be as simple as an 8-bit adder, or it might be able to add, subtract, multiply and divide 8-bit values.
- The test register is a special latch that can hold values from comparisons performed in the ALU. An ALU can normally compare two numbers and determine if they are equal, if one is greater than the other, etc. The test register can also normally hold a carry bit from the last stage of the adder. It stores these values in flip-flops and then the instruction decoder can use the values to make decisions.
- There are six boxes marked "3-State" in the diagram. These are tri-state buffers. A tri-state buffer can pass a 1, a 0 or it can essentially disconnect its output (imagine a switch that totally disconnects the output line from the wire that the output is heading toward). A tri-state buffer allows multiple outputs to connect to a wire, but only one of them to actually drive a 1 or a 0 onto the line.
- The instruction register and instruction decoder are responsible for controlling all of the other components.

64-bit Processors

Sixty-four-bit processors have been with us since 1992, and in the 21st century they have started to become mainstream. Both Intel and AMD have introduced 64-bit chips, and the Mac G5 sports a 64-bit processor. Sixty-four-bit processors have 64-bit ALUs, 64-bit registers, 64-bit buses and so on.

One reason why the world needs 64-bit processors is because of their enlarged address spaces. Thirty-two-bit chips are often constrained to a maximum of 2 GB or 4 GB of RAM access. That sounds like a lot, given that most home computers currently use only 256 MB to 512 MB of RAM. However, a 4-GB limit can be a severe problem for server machines and machines running large databases. Moreover, even home machines will start bumping up against the 2 GB or 4 GB limit pretty soon if current trends continue. A 64-bit chip has none of these constraints because a 64-bit RAM address space is essentially infinite for the foreseeable future - 264 bytes of RAM is something on the order of a quadrillion gigabytes of RAM.

With a 64-bit address bus and wide, high-speed data buses on the motherboard, 64-bit machines also offer faster I/O (input/output) speeds to things like hard disk drives and video cards. These features can greatly increase system performance.

Servers can definitely benefit from 64 bits, but what about normal users? Beyond the RAM solution, it is not clear that a 64-bit chip offers "normal users" any real, tangible benefits now. They can process data (very complex data features lots of real numbers) faster. People doing video editing and people doing photographic editing on very large images benefit from this kind of computing power. High-end games will also benefit, once they are re-coded to take advantage of 64-bit features. Nevertheless, the average user who is reading e-mail, browsing the Web and editing Word documents is not really using the processor in that way. In addition, operating systems like Windows XP have not yet been upgraded to handle 64-bit CPUs. Because of the lack of tangible benefits, it will be 2010 or so before we see 64-bit machines on every desktop.

RAM and ROM

The previous section talked about the address and data buses, as well as the RD and WR lines. These buses and lines connect either to RAM or to ROM - generally both. In our sample microprocessor, we have an address bus 8 bits wide and a data bus 8 bits wide. That means that the microprocessor can address (2^8) 256 bytes of memory, and it can read or write 8 bits of the memory at a time. Let's assume that this simple microprocessor has 128 bytes of ROM starting at address 0 and 128 bytes of RAM starting at address 128.

ROM stands for read-only memory. A ROM chip is programmed with a permanent collection of pre-set bytes. The address bus tells the ROM chip which byte to get and place on the data bus. When the RD line changes state, the ROM chip presents the selected byte onto the data bus.

RAM stands for random-access memory. RAM contains bytes of information, and the microprocessor can read or write to those bytes depending on whether the RD or WR line is signaled. One problem with today's RAM chips is that they forget everything once the power goes off. That is why the computer needs ROM.

By the way, nearly all computers contain some amount of ROM (it is possible to create a simple computer that contains no RAM - many microcontrollers do this by placing a handful of RAM bytes on the processor chip itself - but generally impossible to create one that contains no ROM). On a PC, the ROM is called the BIOS (Basic Input/Output System). When the microprocessor starts, it begins executing instructions it finds in the BIOS. The BIOS instructions do things like test the hardware in the machine, and then it goes to the hard disk to fetch the boot sector. This boot sector is another small program, and the BIOS stores it in RAM after reading it off the disk. The microprocessor then begins executing the boot sector's instructions from RAM. The boot sector program will tell the microprocessor to fetch something else from the hard disk into

RAM, which the microprocessor then executes, and so on. This is how the microprocessor loads and executes the entire operating system.

Microprocessor Performance

The number of transistors available has a huge effect on the performance of a processor. As seen earlier, a typical instruction in a processor like an 8088 took 15 clock cycles to execute. Because of the design of the multiplier, it took approximately 80 cycles just to do one 16-bit multiplication on the 8088. With more transistors, much more powerful multipliers capable of single-cycle speeds become possible. More transistors also allow for a technology called pipelining. In a pipelined architecture, instruction execution overlaps. Therefore, even though it might take five clock cycles to execute each instruction, there can be five instructions in various stages of execution simultaneously. That way it looks like one instruction completes every clock cycle.

Many modern processors have multiple instruction decoders, each with its own pipeline. This allows for multiple instruction streams, which means that more than one instruction can complete during each clock cycle. This technique can be quite complex to implement, so it takes many transistors.

The trend in processor design has primarily been toward full 32-bit ALUs with fast floating-point processors built in and pipelined execution with multiple instruction streams. The newest thing in processor design is 64-bit ALUs, and people are expected to have these processors in their home PCs in the next decade. There has also been a tendency toward special instructions (like the MMX instructions) that make certain operations particularly efficient. There has also been the addition of hardware virtual memory support and L1 caching on the processor chip. All of these trends push up the transistor count, leading to the multi-million transistor powerhouses available today. These processors can execute about one billion instructions per second!

UNIT 2

THE PROGRAMMING PROCESS

I PRE-READING

1. Recollect everything you studied concerning the topic, make notes. Write down the list of words in the Ukrainian language you think you will need while speaking on the topic.

2. a). Read the text and try to guess whether there are words from your list in it.

b). If you don't know the words, translate them using the dictionaries.

c). Choose the words from the list which are the most relevant for this topic.

d). If you don't know the meaning of any words or abbreviations, find their explanations in the Glossary.

e). Which words from exercises 1, 2c may be considered as general notions and which ones as technical terms?

f). Give explanations of several terms, using the Glossary.

Programming is instructing a computer to do something for you with the help of a programming language. The role of a programming language can be described in two ways:

Technical: It is a means for instructing a Computer to perform Tasks

Conceptual: It is a framework within which we organize our ideas about things and processes.

According to the last statement, in programming we deal with two kinds of things: data, representing ``objects" we want to manipulate procedures, i.e. ``descriptions" or ``rules" that define how to manipulate data.

A programming language should both provide means to describe primitive data and procedures and means to combine and abstract those into ones that are more complex.

The distinction between data and procedures is not that clear cut. In many programming languages, procedures can be passed as data (to be applied to ``real" data) and sometimes processed like ``ordinary" data. Conversely ``ordinary" data can be turned into procedures by an evaluation mechanism.

There are very few essential elements one needs for programming:

- variables (things to store information)
- procedural abstractions (named collections of instructions)
- branching (IF's)
- side-effects (``Set", ``="), to assign values to variables procedure/function calls loops (as ``cheap" alternative to procedure calls)

Those major elements rely on a number of things like:

- - primitive data structures, e.g. numbers and symbols
- input/output functions
- libraries of useful procedures and functions

To make a programming language useful, you also need things like:

- - Ways to build compound data structures (e.g. lists, arrays, records, objects)
- Complex conditions and several kinds of loops
- - Error handling mechanisms

II READING

1. Read and translate the text into Ukrainian.

A programmer must tell the computer how to solve a problem step by step. Programmers should strive to write high-quality, reliable, efficient, maintainable, and easy-to-read-and-understand programs that allow a minimum number of input errors. Following several steps ensures a well-written program. First, the programmer should define the problem according to the user's needs. Next, the programmer must design the solution, including required inputs, outputs, and basic logic patterns. The remaining steps writing, compiling, debugging, and testing-involve coding the solution in the appropriate computer language.

Most programmers agree that using structured programming techniques will best meet the criteria of well-written software with a minimum of time and costs. Four methodologies characterize structured programming: top-down design, program documentation, program testing, and use of a programming team.

Top-down design employs the modular approach, breaking a problem into smaller and smaller subproblems, each written as a separate module. Program documentation should occur at all programming stages, from defining the problem to debugging and testing the program. Program testing should also occur at all stages in order to isolate errors and reduce development time and costs. In a chief programmer, team (CPT) a chief programmer heads a team of programmers whose goals are the goals of structured programming: to produce software that is easy to understand, maintain and modify; to decrease costs; and to increase program reliability.

Defining a programming problem begins with recognizing the need for information. The definition should include a statement of the problem, the needed output, and the required input. The definition should be mutually acceptable to all parties involved in the programming effort, including the user.

A programmer must understand the four problem-solving logic patterns to plan a programming solution: simple sequence, selection, loop, and branch. (The branch is

not considered an option in structured programming.) Design aids help a programmer avoid errors and omissions that might occur in a less structured programming method. Design aids include pseudocode, flowcharts, structure charts, HIPO charts, and Nassi-Shneiderman Charts.

Once the programming solution is planned, the programmer chooses an appropriate language and writes the program. A programmer using structured coding follows four rules: : aid the GOTO statement, limit the size of the modules to no more than 50 or 60 lines of code, construct a proper program module that has only one entry and one exit, and include liberal documentation throughout the program.

Several types of program statements are common to most high-level programming languages. Comments, or remark statements, have no effect on program execution. Declaration, input-output, computation, transfer or control, and comparison statements all affect program execution.

After writing a program, a programmer must submit the program to the computer for translation. The programmer uses a sequence of instructions written in one of three programming language levels: machine language, assembly language, or high-level language. A computer can understand only machine language without interpretation. A language-translator program translates source programs written in either assembly or high-level language into a machine-executable form known as the object program.

Programs must be debugged and tested. Desk-checking and structured walkthrough methods help programmers debug and test programs manually. Programmers use sample data and compare computer-determined output with manually prepared correct results.

Program maintenance refers to the revisions needed to correct, improve, update, or expand programs. Structured programming and proper documentation help programmers maintain programs efficiently.

Standard operating systems allow the use of a wider variety of software products. Improved user interface has helped users take advantage of the many types of software available for microcomputers. Interface describes the way in which a user submits data to a computer and receives information from it. Types of interfaces include command-line interfaces, icons, and menus.

2. Make up the logical scheme of the text and render the content of the text based on the scheme.

3. Fill in the gaps with the words from the list.

a. structured walkthrough f. pseudocode

b. visual table of contents
c. chief programmer team (CPT)
d. structured programming
e. modular approach
g. interfaces
h. branch
i. language translation program
j. flowchart

1. A standard design procedure called _____encourages programmers to think about a programming problem first.

2. Top-down design employs the _____, breaking a problem into smaller and smaller subproblems.

3. A coordinated programming effort involves formation of a _____to produce efficient programs in a minimum amount of time.

4. A command often signaled by GOTO, the _____pattern, is the most controversial programming pattern.
5. _____ is a narrative description of processing steps in a program.
6. A block diagram, or _____, provides a visual frame of reference to program processing steps.
7. The most general level of diagram, the _____, looks similar to a structure chart.
8. A _____ transforms the source program into an object program.
9. A programming team member may conduct a _____ in the debugging process to minimize overall development costs.

10. Types of _____ include command lines, icons, and menus.

4. Translate the sentences into the Ukrainian language. Take into consideration the information presented in the sentences. It may be helpful to you while doing the next exercises and discussing the topic.

1. A modular approach also makes testing, debugging, and updating more efficient since reviewing one subsection at a time is easier than reviewing an entire program at one time. 2. Pseudocode allows a programmer to focus on the steps required to perform a particular process, rather than on the syntax of a particular computer language; and it allows a user to understand the purpose and function of each module of a program. 3. If an unanticipated condition arises during processing, the program will be executed incorrectly or not at all. 4. Proper program documentation is a reference guide for programmers and analysts who must modify a program. 5. Defining and designing are the most difficult steps because the programmer must define a problem according to the user's needs and design it according to the computer's capabilities. 6. Conditional transfers and comparisons depend on a condition or result of a comparison to transfer control to another portion of a program. 7. The programmer selects the programming language that best fits the processing requirements of the program. 8. Different kinds of errors can occur at the various steps of program development. 9. Structured programming techniques

encourage programmers to think about the problem first rather than spend an unreasonable amount of time debugging later. 10. Pseudocode, flowcharts, structure charts, HIPO charts, and Nassi-Shneiderman Charts are design aids which provide ways to illustrate algorithms visually and should be included with program documentation.

5. Choose the right answer.

1. Unless otherwise instructed, a computer will execute program steps_____.
   a. logically
   b. in sequential order
   c. incorrectly
   d. in modules

2. Programmers agree that a successful software program should be_____.
   a. reliable, error free, maintainable, efficient, and readable
   b. written by a system analyst
   c. based on Nassi-Shneiderman charts
   d. written by a programmer

3. A programmer should follow four steps in problem solving and program development. They are _____.
   a. document, write, design, debug
   b. compile, debug, test, revise
   c. define, document, design, document
   d. define, design, write, compile

4. A computer translates a program code into_____.
   a. structured program
   b. pseudocode
   c. machine language
   d. a source program

5. Techniques designed to eliminate logic errors and design, maintenance, and debugging time are called _____.
   a. structured programming techniques
   b. Nassi-Shneiderman charts
   c. HIPO techniques
   d. structured flowcharts

6. A method of programming which proceeds from the general to the specific is called _____.
   a. modulating
   b. top-down design
   c. documentation designing
   d. structure charts

7. Program explanations, descriptions of modules, and design charts are all a part of _____.
   a. pseudocode
   b. documentation
   c. the source program
   d. syntax

8. Documentation and _____ should occur at each problem-solving step.
   a. debugging
   b. teamwork
   c. program testing
   d. defining

9. Before a programmer proceeds, the problem must be clearly and mutually defined _____.

a. according to accepted standards
b. by the user
c. by the chief programming team
d. by all parties involved

10. Information requirements of a user become clear by defining required _____.

a. input
b. problem parameters
c. output
d. programming language

11. A shift of control of computer execution can occur in two ways. One way, a trailer value, signals the computer to stop performing a(n) _____.

a. loop
b. sequence
c. execution
d. computation

12. Simple sequence, selection, loop, and branch are four basic_____.

a. design features
b. logic patterns
c. documentation requirements
d. modules

13. A programming team can visualize the program design and suggest changes with the help of _____.

a. flowlines
b. design aids
c. a language translator
d. the object program

14. What type of statement always changes the sequence of execution?

a. comment
b. input
c. unconditional transfer
d. comparison

15. Programmers sometimes trade their partially debugged programs among themselves. This type of peer review is called a _____.

a. desk check
b. top-down check
c. structured review
d. structured walkthrough

6. Translate the words and word-combinations into English.

Мова асемблера, оператор програми, документація до задачі, структурна блок-схема, низхідне проектування, інтерфейс командних рядків, обчислення, налагоджування, оператор переходу, фіктивний модуль, виконавча форма, засоби проектування, лінія зв'язку, графічний символ, ремонтопридатність, вихідна програма, псевдокод, початкова програма, структурна схема, структурний критичний аналіз.

7. Say true or false and explain.

1. T F Programming methods have gone from haphazard to procedural.
2. T F A programmer must first design and document the solution.

3. T F Programming methodologies are often referred to as the structured approach to programming.
4. T F Structured programming encourages programmers to think about a problem first and reduces debugging time.
5. T F Top-down design employs the structured approach to define the major steps or functions of a program.
6. T F Breaking each major step into smaller steps helps refine a module.
7. T F The documentation portion of a program explains the steps of a program to users and programmers.
8. T F Computer instructions are based on four basic logic patterns: assembler, compiler, interpreter, and translator.
9. T F Desk-checking describes a method of reviewing another programmer's work.
10. T F Standard operating systems allow the use of a wider variety of software products.

III DISCUSSING

1. Give the short answer.

1. List five program qualities necessary for successful software.
2. Why is defining the problem the most difficult programming step?
3. State several reasons why structured problem solving is crucial.
4. Name a common type of documentation statement and state its purpose.
5. List the processes that should occur at every step of program development
6. List the three types of information that should be included in the documentation defining the problem.
7. Explain how a modular approach is important to program design.
8. List the four basic logic patterns upon which computer instructions are based.
9. What is the advantage of using pseudocodes?
10. Explain the interaction between language-translator, source, and object programs.

2. Prepare reports using Internet or other sources (catalogues, magazines, books, etc.) about the latest news, achievements in the field concerning the topic of the chapter.

3. Summarize everything you have learnt on the topic.

4. Answer the question, taking into consideration quick pace of the latest achievements and developments in computer engineering and programming and information presented in the reports, what information from the texts of the Chapter should be reconsidered, changed, added, etc…

5. It's interesting to know…

Programming Languages

Programming languages contain the series of commands that create software. In general, a language that is encoded in binary numbers or a language similar to binary numbers that a computer's hardware understands is understood more quickly by the computer. A program written in this type of language also runs faster. Languages that use words or other commands that reflect how humans think are easier for programmers to use, but they are slower because the language must be translated first so the computer can understand it.

Machine Language

Computer programs that can be run by a computer's operating system are called executables. An executable program is a sequence of extremely simple instructions known as machine code. These instructions are specific to the individual computer's CPU and associated hardware; for example, Intel Pentium and Power PC microprocessor chips each have different machine languages and require different sets of codes to perform the same task. Machine code instructions are few in number (roughly 20 to 200, depending on the computer and the CPU). Typical instructions are for copying data from a memory location or for adding the contents of two memory locations (usually registers in the CPU). Machine code instructions are binary—that is, sequences of bits (0s and 1s). Because these numbers are not understood easily by humans, computer instructions usually are not written in machine code.

Assembly Language

Assembly language uses commands that are easier for programmers to understand than are machine-language commands. Each machine language instruction has an equivalent command in assembly language. For example, in assembly language, the statement "MOV A, B" instructs the computer to copy data from one location to another. The same instruction in machine code is a string of 16 0s and 1s. Once an assembly-language program is written, it is converted to a machine-language program by another program called an assembler. Assembly language is fast and powerful because of its correspondence with machine language. It is still difficult to use, however, because assembly-language instructions are a series of abstract codes. In addition, different CPUs use different machine languages and therefore require different assembly languages. Assembly language is sometimes inserted into a high-level language program to carry out specific hardware tasks or to speed up a high-level program.

High-Level Languages

High-level languages were developed because of the difficulty of programming assembly languages. High-level languages are easier to use than machine and assembly languages because their commands resemble natural human language. In addition, these languages are not CPU-specific. Instead, they contain general commands that work on different CPUs. For example, a programmer writing in the high-level Pascal programming language who wants to display a greeting need include only the following command:

Write (‘Hello, User!’);

This command directs the computer’s CPU to display the greeting, and it will work no matter what type of CPU the computer uses. Like assembly language instructions, high-level languages also must be translated, but a compiler is used. A compiler turns a high-level program into a CPU-specific machine language. For example, a programmer may write a program in a high-level language such as C and then prepare it for different machines, such as a Cray Y-MP supercomputer or a personal computer, using compilers designed for those machines. This speeds the programmer’s task and makes the software more portable to different users and machines.

American naval officer and mathematician Grace Murray Hopper helped develop the first commercially available high-level software language, FLOW-MATIC, in 1957. Hopper is credited for inventing the term bug, which indicates a computer malfunction; in 1945, she discovered a hardware failure in the Mark II computer caused by a moth trapped between its mechanical relays. From 1954 to 1958 American computer scientist John Backus of International Business Machines, Inc. (IBM) developed FORTRAN, an acronym for FORmula TRANslation. It became a standard programming language because it can process mathematical formulas. FORTRAN and its variations are still in use today. Beginner’s All-purpose Symbolic Instruction Code, or BASIC, was developed by American mathematician John Kemeny and Hungarian-American mathematician Thomas Kurtz at Dartmouth College in 1964. The language was easier to learn than its predecessors were and became popular due to its friendly, interactive nature and its inclusion on early personal computers (PCs). Unlike other languages that require that all their instructions be translated into machine code first, BASIC is interpreted—that is, it is turned into machine language line by line as the program runs. BASIC commands typify high-level languages because of their simplicity and their closeness to natural human language. For example, a program that divides a number in half can be written as

```
10 INPUT “ENTER A NUMBER,” X
20 Y=X/2
30 PRINT “HALF OF THAT NUMBER IS,” Y
```

The numbers that precede each line are chosen by the programmer to indicate the sequence of the commands. The first line prints “ENTER A NUMBER” on the computer screen followed by a question mark to prompt the user to type in the number labelled “X.” In the next line, that number is divided by two, and in the third line, the result of the operation is displayed on the computer screen.

Other high-level languages in use today include C, Ada, Pascal, LISP, Prolog, COBOL, HTML, and Java. New compilers are being developed, and many features available in one language are being made available in others.

Object-Oriented Programming Languages

Object-oriented programming (OOP) languages like C++ are based on traditional high-level languages, but they enable a programmer to think in terms of collections of cooperating objects instead of lists of commands. Objects, such as a circle, have properties such as the radius of the circle and the command that draws it on the computer screen. Classes of objects can inherit features from other classes of objects. For example, a class defining squares can inherit features such as right angles from a class defining rectangles. This set of programming classes simplifies the programmer’s task, resulting in more reliable and efficient programs.

Overall Structure

Programming languages can be implemented with compilers, assemblers, and interpreters. A compiler converts high-level source code into executable object code (also called binary code or machine code). An assembler converts assembly language source code into executable object code. An interpreter translates either source code or tokens into machine code, one instruction at a time, as the program is run. Object oriented programming (as well as the implementation of some kinds of data structures) blurs the line between compilers and interpreters.

Nature: The two most common kinds of programming languages are procedural and object oriented.

Format: The two basic formats for programming languages are free form and column. Older programming languages were typically designed to fit into the columns of Hollerith "punch" cards, with specific fields being reserved for specific purposes. Modern programming languages are free form in the sense that there are no specific columns that various items must line up in.

Lexical Elements

Character set: There are two character sets to consider the source code character set and the execution character set. The source code character set is the set of characters allowed for writing source code programs. The execution character set is the set of characters recognized and used by running programs. These may be the same character set, but usually are different (usually there are

more characters allowed in the execution character set and some languages have mechanisms for specifically handling multiple execution character sets. The source code character set can vary for different parts of the language. Some programming languages have case sensitive source code character sets (that is, identifiers that differ only in case are considered to be different identifiers), while other programming languages have case insensitive source code character sets (that is, identifiers that differ only in case are considered to be the same identifier). Very few programming languages require a specific character encoding for the execution character set. Some of the common character encodings in use include ASCII, EBCDIC, and Unicode. A common cross-platform programming error is to assume a specific character set. For example, the expression ('Z[1] - 'A' + 1) will produce the number of letters in the alphabet (26) when either ASCII or Unicode is used, but will produce the answer 41 when EBCDIC is used (because the letters of the alphabet are not encoded consecutively in EBCDIC). Another common error regards the use of the high eight-bit characters in ASCII, which are typically different on each different operating system.

White space: White space is any blanks (or space characters) in the source code. Some programming languages define additional characters as white space. These may include horizontal tab, vertical tab, end of line, form feed, carriage return, and line feed. In some programming languages, comments are treated as white space. White space is sometimes compacted into a single space character by the compiler. White space may be optional or required (and whether it is optional or required can vary at different parts of the source code).

Line termination: Some modern programming languages ignore line returns, while others use them as dividing lines between instructions (sometimes with a special character to indicate that an instruction continues on more than one line and sometimes with a special character to allow more than one instruction per line).

Line length: Some programming languages have fixed line lengths, while others have variable line lengths. Fixed line lengths typically occur with programming languages that use columns.

Multibyte characters: Some programming languages have special multibyte encodings for some characters. It is common to have an escape character(s) to allow programmers to indicate special characters (such as non-printable characters, line breaks, characters' reserved as special symbols, etc.) in character or string constants. Some programming languages have special character sequences for encoding required characters on older hardware that doesn't support all of the required source code character set. Some programming languages include have special sequences for encoding multi-byte characters (such as non-Roman alphabets).

Comments: Every programming language except for machine language (writing in raw binary opcodes) has some system for comments (and comments

are sometimes even interspersed as ASCII data between raw binary machine code). Comments are used to make programs self-documenting for future human maintenance or modification (most of the life of any successful software is in maintenance, not initial programming). Comments are either removed or replaced with white space.

Tokens: Tokens are the basic lexical building blocks of source code. Characters are combined into tokens according to the rules of the programming language. There are five classes of tokens: identifiers, reserved words, operators, separators, and constants.

Control structures: There are three basic kinds of control structures: sequences, branching, and loops.

Sequential structures: Sequential structures are structures that are stepped through sequential. These are also called sequences or iterative structures. Basic arithmetic, logical, and bit operations are in this category. Data moves and copies are sequences. Most register and processor control operations are sequences. Some computer scientists group subroutines and functions as a special kind of sequence, while others group them as a special kind of branching structure.

Branching structures: Branching structures consist of direct and indirect jumps (including the infamous "GOTO"), conditional jumps (IF), nested ifs, and case (or switch) structures.

Loop structures: The basic looping structures are DO iterative, do WHILE, and do UNTIL. An infinite loop is one that has no exit. Normally, infinite loops are programming errors, but event loops and task schedulers are examples of intentional infinite loops.

Objects: Object oriented programming does not fit neatly into the three basic categories, because the three basic categories of control structures describe procedural programming. The internal implementation of at least some methods will use the three basic structures, but this is hidden from other objects.

Machine code: Machine code can be written using the three basic structures, although on some particular processors it may not be possible to write all of the structures as atomic code. Atomic means all as one unit. On some processors, it may be necessary to construct some structures with a series of machine instructions instead of with a single machine instruction.

Exception processing: Exception processing also doesn't fall into the three basic categories. Normally, exception processing only occurs in low-level assembly or machine language programming. Some high-level languages provide "on conditions", or the ability to designate routines to be run when an exception or other special event occurs. The actual code that runs during the exception can be written with just the three basic structures.

UNIT 3

SOFTWARE

I PRE-READING

1. Recollect everything you studied concerning the topic, make notes. Write down the list of words in the Ukrainian language you think you will need while speaking on the topic.

2. a). Read the text and try to guess whether there are words from your list in it.

b). If you don't know the words, translate them using the dictionaries.

c). Choose the words from the list which are the most relevant for this topic.

d). If you don't know the meaning of any words or abbreviations, find their explanations in the Glossary.

e). Which words from exercises 1, 2c may be considered as general notions and which ones as technical terms?

f). Give explanations of several terms, using the Glossary.

Twenty years ago, less than 1 percent of the public could have intelligently described what "computer software" meant. Today, most professionals and many members of the public at large feel they understand software. However, do they?

A textbook description of software might take the following form:

> ***Software:*** (1) instructions (computer programs) that when executed provide desired function and performance; (2) data structures that enable the programs to adequately manipulate information; (3) documents that describe the operation and use of the programs.

There is no question other, more complete definitions could be offered, but we need more than a formal definition.

To gain an understanding of software (and ultimately of software engineering), it is important to examine the characteristics of software that make it different from other things that human beings build. When hardware is built, the human creative process (analysis, design, construction, testing) is ultimately translated into a physical form. If we build a new computer, our initial sketches, formal design drawings, and bread boarded prototype evolve into a physical product with VLSI chips, circuit boards, power supplies, etc.

Software is a *logical* rather than *a physical* system element.

II READING

1. Read and translate the text into Ukrainian.

There are two basic types of computer programs: system programs and application programs. System programs directly affect the operation of the computer; they are designed to optimize use of hardware and help the computer manage its own resources. Operating systems, utility programs, and programming language translators are examples of system programs. Application programs are used to perform specific data processing or computational tasks that solve an organization's information needs or help complete an educational or personal task. Examples of application programs include word processors, data management packages, spreadsheets, and graphics packages.

Early computer systems required human operators to monitor computer operations, determine the order in which programs were to be run, and prepare the input and output devices. While advances were made in hardware, the speed and unreliability of humans remained unchanged. In the 1960s, the development of operating systems eliminated many of the problems associated with human operators.

An operating system provides an interface between application programs and the hardware of the computer. It increases the efficiency of computer operations by eliminating human intervention, allowing several programs to share computer resources, and scheduling jobs on a priority basis.

Even with the development of operating systems, early computers were still not efficient. The use of multiprogramming, virtual storage, and multiprocessing further increased system efficiency. Multiprogramming frees the computer to work with more than one program and allows many programs to reside in primary storage concurrently. The CPU executes instructions in one program and switches to another program when efficient to do so.

Virtual storage makes main storage appear larger than it actually is. Virtual storage is based on the principle that only the part of the program that is needed is kept in primary storage. The remaining part of the program is kept in secondary storage. Since only a fragment of the program is kept in primary storage, more programs can be kept in primary storage simultaneously. This allows more programs to be executed within a given time period.

Multiprocessing occurs when two or more CPUs are linked together and their actions coordinated. Multiprocessing systems are designed to achieve a specific function. Many times a small CPU will handle editing of incoming data and free the large CPU for more important tasks. In a case like this, the small CPU is known as a front-end processor. A small CPU can also be used as the interface between the large CPU and a database. In this case, the small CPU might access data. This is known as a back-end processor.

Early computers were programmed by physically arranging wires and switches. A code, called machine language, was developed to correspond to the on and off electrical states needed to enter instructions to these computers. Machine and assembly languages are known as low-level languages; they give the programmer control over computer hardware, memory locations, and I/O operations. Machine language uses Is and Os to program the computer. To overcome the difficulty of working with 1s and Os, assembly language was developed. Mnemonics are used to specify instructions in assembly. An advantage of working with low-level languages is speed of execution.

The first high-level language was FORTRAN. The difference between a high- and low-level language is that the low-level language is machine oriented requiring the programmer to have detailed knowledge of how the computer works, whereas the high-level language is designed to let the programmer concentrate on problem solving.

Natural (query) languages use Englishlike sentences to retrieve information usually contained in a database. Even though natural languages were designed for the novice computer user, they are very powerful and useful. In the past, natural languages were limited to use with mainframes, but natural language systems are now being developed for use with smaller computers.

There are several factors to consider when choosing a language. What languages does your computer system support? What type of processing will take place? Are equipment changes planned? Because of the differences in languages, many organizations use more than one language.

Software companies lose millions of dollars each year because of illegal duplication of software. To combat piracy, software is copyrighted, giving the copyright owner specific legal rights and allowing the owner to sue any violators. Users are allowed to make only one archival copy of their software. Many software firms are using protection codes to discourage illegal copying. The protection code prevents unauthorized duplication. There are programs available that break the protection codes and individuals are using these programs to make copies that are then sold or passed along to friends. Is the answer better protection codes? As soon as a new protection scheme comes on the scene, there are people who are willing to circumvent the copy protection. Only a change in ethics will begin to solve the piracy problem.

Retell the most important information from the text.

2. Make up the logical scheme of the text and render the content of the text based on the scheme.

3. Fill in the gaps with the words from the list.

a. graphics
b. IF THEN
c. mnemonic
g. GOTO
h. protection codes
i. operand

d. thrashing
e. interactive
f. multiprogramming
j. segmentation
k. online
1. utility programs

1. One of the major categories of application programs is _____.
2. An interactive home banking system is an example of _____ processing.
3. Operating systems include processing programs called _____that perform specialized functions.
4. When the CPU executes instructions from more than one program separately, but at speeds that make the programs appear to be executing simultaneously, it is referred to as _____.

5. One method of implementing virtual storage is _____.

6. If virtual storage is not used wisely, much time can be spent locating and exchanging segments and little actual processing occurs. This is called _____.

7. Each machine language instruction has an op code and a(n) _____.

8. A language that allows the programmer to communicate directly with the computer during programming is known as a(n) _____language.

9. The main criticism of BASIC focuses on the use of the _____statement.

10. A popular method used by software developers to discourage piracy is _____.

4. Translate the sentences into the Ukrainian language. Take into consideration the information presented in the sentences. It may be helpful to you while doing the next exercises and discussing the topic.

1. Operating system programs are stored on a secondary storage device called the system residence. 2. In order to write low-level language programs, a programmer must possess a great deal of technical knowledge about the microprocessor used by the computer and be very detail oriented. 3. Because Pascal was developed after the advent of structured programming, it is designed to support structured programming. 4. FORTRAN, the oldest high-level language, was developed in the mid- 1950s and was used primarily in scientific and mathematical circles because of its mathematical capabilities. 5. Without software to take advantage of the new capabilities of computers, the full benefit of the computer as a problem-solving tool will not be realized. 6. Library programs are stored in the system library usually on tape or disk. 7. The librarian program manages the system library. 8. The operating system is used by the computer to manage its own operations and provides an interface between the user and the computer hardware. 9. When two or more CPUs are linked together for coordinated operation, this is called multiprocessing. 10. Low-level languages enable the programmer to specify memory locations, direct input and output operations, and govern the use of

registers; and high-level languages allow the programmer to focus on problem solving rather than machine oriented issues

5. Choose the right answer.

1. The benefits of smaller, faster, and more powerful computers cannot be realized without the existence of well-prepared _____.
   a. keyboards
   b. hardware
   c. software
   d. analog switches

2. A(n) _____ solves particular user problems.
   a. application program
   b. flowchart
   c. system program
   d. binary algorithm

3. _____ coordinate the operation of computer circuitry.
   a. Voltage spikes
   b. Magnetic fields
   c. System programs
   d. Disk drives

4. The_____ provides an interface between the user or application program and the computer hardware.
   a. disk drive controller
   b. CPU
   c. interface optimizer card
   d. operating system

5. The _____ is the first program to be transferred into primary storage.
   a. operating system
   b. supervisor program
   c. job-control program
   d. application program

6. So that library programs will not have to be rewritten every time they are needed, they are stored in a _____.
   a. supervisor library
   b. local library
   c. memory location
   d. system library

7. When a computer seems to work with more than one program, this is called _____.
   a. processing efficiency
   b. multiprogramming
   c. bi-computing
   d. multiple program tasking

8. When only the immediately needed portion of a program is kept in primary storage, this is called _____.
   a. multiprogramming
   b. memory allocation
   c. virtual storage
   d. virtual memory allocation

9. _____involves the use of two or more CPUs linked together for coordinated operation.
   a. Multiprocessing
   b. Multiprogramming
   c. Multiple program tasking
   d. Thrashing

10. What kind of language gives the programmer direct control over details of computer hardware?

a. Low-level
b. BASIC
c. Interpreted
d. Compiled

11. A difference between low-level languages and high-level languages is that low-level languages are machine oriented and high-level languages focus on _____.

a. resource allocation
b. input and output operations
c. problem solving
d. program syntax

12. The oldest high-level programming language is_____.

a. COBOL
b. machine
c. FORTH
d. FORTRAN

13. A disadvantage of COBOL is that it is_____.

a. self-documenting
b. wordy
c. Englishlike
d. standardized

14. The language commonly associated with artificial intelligence is _____.

a. assembly language
b. C
c. Ada
d. LISP

15. Programming languages that use Englishlike sentences to access information usually contained in a database are called _____.

a. high-level languages
b. data base management systems
c. middle-level languages
d. natural (query) languages

6. Translate the words and word-combinations into English.

Генератор звітів, електронна таблиця, службова програма, архівна копія, нерезидентна програма, переповнення (пам'яті), редактор зв’язків, керування пам'яті, сторінкова організація, підпрограма, розмовна мова/мова запитів, прикладна програма, попередній процесор, адміністративна система введення/виведення , мова керування завданнями, бібліотекар, процесор бази даних, авторське право, редагування, програма опрацювання.

7. Say true or false and explain.

1. T F The two basic types of programs are system programs and application programs.
2. T F Program processing can occur in two ways, either by batch or serial processing.
3. T F Control programs and processing programs are types of programs that make up the operating system.

4. T F A job-control language is the communication link between the programmer and the operating system.
5. T F A language-translator program translates the object program into the source program.
6. T F In multiprogramming, programs are kept separated by using paging.
7. T F A front-end processor is an interface between a large CPU and a data base.
8. T F Assembly language does not use mnemonics.
9. T F When you buy software, you are permitted to make one archival copy.
10. T F The best protection against software piracy is a high standard of personal ethics.

III DISCUSSING

1. Give the short answer.

1. Explain what an operating system is.
2. What is the purpose of an operating system?
3. What are the components of an operating system and where are they usually stored?
4. Briefly explain the principle of virtual storage.
5. What are the two main methods of implementing virtual storage systems?
6. What is thrashing?
7. In what two areas must a programmer be proficient in order to write low-level language programs?
8. Explain the correspondence between an assembly instruction and a machine instruction. Is this relationship beneficial?
9. Why is Pascal considered a good language for beginners to learn?
10. What are two methods of dealing with software piracy?

2. Prepare reports using Internet or other sources (catalogues, magazines, books, etc.) about the latest news, achievements in the field concerning the topic of the chapter.

3. Summarize everything you have learnt on the topic.

4. Answer the question, taking into consideration quick pace of the latest achievements and developments in computer engineering and programming and information presented in the reports, what information from the texts of the Chapter should be reconsidered, changed, added, etc…

5. It's interesting to know…

The Evolving Role of Software

The context in which software has been developed is closely coupled to four decades of computer system evolution. Better hardware performance, smaller size, and lower cost have precipitated more sophisticated computer-based systems. During the early years of computer system development, hardware underwent continual change while software was viewed by many as an after-thought. Computer programming was a seat-of-the-pants art for which few systematic methods existed. Software development was virtually unmanaged unless schedules slipped or costs began to escalate. During this period, a batch orientation was used for most systems. Notable exceptions were interactive systems such as the early American Airlines reservation system and real-time defense-oriented systems such as SAGE. For the most part, however, hardware was dedicated to the execution of a single program that in turn was dedicated to a specific application. During the early years general-purpose hardware became commonplace. Software, on the other hand, was custom designed for each application and had a relatively limited distribution. Product software (i.e., programs developed to be sold to one or more customers) was in its infancy. Most software was developed and ultimately used by the same person or organization. The same person wrote it, got it running, and if it failed, fixed it. Because job mobility was low, managers could rest assured that this person would be there when bugs were encountered. Because of this personalized software environment, design was an implicit process performed in one's head and documentation was often nonexistent. Throughout the early years, we learned much about the implementation of computer-based systems, but relatively little about computer system engineering. In fairness, however, we must acknowledge the many outstanding computer-based systems that were developed during this era. Some of these remain in use today and provide landmark achievements that continue to justify admiration.

The second era of computer system evolution spanned the decade from the mid-1960s to the late 1970s. Multiprogramming, multi-user systems introduced new concepts of human-machine interaction. Interactive techniques opened a new world of applications and new levels of hardware and software sophistication. Real-time systems could collect, analyze, and transform data from multiple sources, thereby controlling processes and producing output in milliseconds rather than minutes. Advanced in on-line storage devices led to the first generation of database management system. The second era was also characterized by the use of product software and the advent of "software houses." Software was developed for widespread distribution in a multidisciplinary market. Programs for mainframes and minicomputers were distributed to hundreds and sometimes thousands of users. Entrepreneurs from

industry, government, and academia broke away to "develop the ultimate software package" and earn a bundle of money. As the number of computer-based systems grew, libraries of computer software began to expand. In-house developed projects produced tens of thousands of program source statements. Software products purchased from the outside-added hundreds of thousands of new statements. A dark cloud appeared on the horizon: All of these programs—all of these source statements— had to be corrected when faults were detected, modified as user requirements changed, or adapted to new hardware that was purchased. These activities were collectively called software maintenance. Effort spent on software maintenance began to absorb resources at an alarming rate. Worse yet, the personalized nature of many programs made them virtually unmaintainable. A software crisis had begun.

The third era of computer system evolution began in the mid-1970s and continues today. The distributed system—multiple computers, each performing functions concurrently and communicating with one another—greatly increased the complexity of computer-based systems. Global and local area networks, high bandwidth digital communications, and increasing demands for "instantaneous" data access put heavy demands on software developers. The third era has also been characterized by the advent and widespread use of microprocessors and personal computers. The microprocessor is an integral part of a wide array of "intelligent" products including automobiles, microwave ovens, industrial robots, and blood serum diagnostic equipment. In many cases software technology is being integrated into products by technical staff who understands hardware but are often novices in software development. The personal computer has been the catalyst for the growth of many software companies. While the software companies of the second era sold hundreds or thousands of copies of their programs, the software companies of the third era sell tens and even hundred of thousands of copies. Personal computer hardware is rapidly becoming a commodity, while software provides the differentiating characteristics. In fact, as the rate of personal computer sales growth flattened during the mid-1980s, software product sales continued to grow. Many people in industry and at home spent more money on software than they did to purchase the computer on which the software runs.

The fourth era in computer software is just beginning. Authors such as Feign Baum and McCorduck predict that "fifth generation" computers and their related software will have a profound impact on the balance of political and industrial power throughout the world. Already, fourth generation techniques for software development are changing the manner in which some segments of the software community build computer programs. Expert systems and artificial intelligence

software have finally moved from the laboratory into practical application for wide-ranging problems in the real world.

As we move into the fourth era, the software crisis continues to intensify. For now, we can describe the software crisis in the following manner:

1. Hardware sophistication has outpaced our ability to build software that can tap hardware's potential. 2. Our ability to build new programs cannot keep pace with the demand for new programs. 3. Our ability to maintain existing programs is threatened by poor design and inadequate resources.

An Industry Perspective

In the early days of computing, computer-based systems were developed using hardware-oriented management techniques. Project managers focused on hardware because it was the single largest budget item for system development. To control hardware costs managers instituted formal controls and technical standards. They demanded thorough analysis and design before something was built. They measured the process to determine where improvement could be made. Stated simply, they applied the controls, methods, and tools that we recognize as hardware engineering. Sadly, software was often little more than an afterthought. In the early days, programming was viewed as an art form. Few formal methods existed and fewer people used them. The programmer often learned his or her craft by trial and error. The jargon and challenges of building computer software created a mystique that few managers cared to penetrate. The software world was virtually undisciplined—and many practitioners of the day loved it! Today, the distribution of costs for the development of computer-based systems has changed dramatically. Software, rather than hardware, is often the largest cost item. Software engineering is the profession which is likely to replace programming and systems analysis over the next ten years. The objectives of software engineering are the development of very large, complex software item, which satisfy strict standards of performance and correctness, in a controlled, scheduled, budgeted and cost-effective way. Software engineers require, in addition to a proficiency in programming, knowledge of formal mathematics and logic, computing science, economics and management. Software engineering is carried out by teams of people. When a software development project is started, the teams are set up in a management structure corresponding to the structure of the software itself. A schedule is drawn up for the project, and costs are reallocated to the various portions and stages. Each team has a team leader, whose task is to make sure that the software developed by the team is correct, properly structured, has the right interfaces, and is on schedule and within budget. This is a very difficult task, which requires a wide range of technical and management skills.

UNIT 4

APPLICATION SOFTWARE

I PRE-READING

1. Recollect everything you studied concerning the topic, make notes. Write down the list of words in the Ukrainian language you think you will need while speaking on the topic.

2. a). Read the text and try to guess whether there are words from your list in it.

b). If you don't know the words, translate them using the dictionaries.

c). Choose the words from the list which are the most relevant for this topic.

d). If you don't know the meaning of any words or abbreviations, find their explanations in the Glossary.

e). Which words from exercises 1, 2c may be considered as general notions and which ones as technical terms?

f). Give explanations of several terms, using the Glossary.

Application software is a subclass of computer software that employs the capabilities of a computer directly to a task that the user wishes to perform. This should be contrasted with system software, which is involved in integrating a computer's various capabilities, but does not directly apply them in the performance of tasks that benefit the user.

A simple, if imperfect, analogy in the world of hardware would be the relationship of an electric light — an application — to an electric power generation plant — the system. The power plant merely generates electricity, itself not really of any use until harnessed to an application like the electric light, which performs a service that the user desires.

Multiple applications bundled together as a package are sometimes referred to as an application suite. Microsoft Office, which bundles together a word processor, a spreadsheet, and several other discrete applications, is a typical example. The separate applications in a suite usually have a user interface that has some commonality making it easier for the user to learn and use each application. And often they may have some capability to interact with each other in ways beneficial to the user. For example, a spreadsheet might be able to be embedded in a word processor document even though it had been created in the separate spreadsheet application.

User software tailors systems to meet the users specific needs. User software includes spreadsheet templates, word processor macros, scientific simulations, and graphics and animation scripts. Even email filters are a kind of user

software. Users create this software themselves and often overlook how important it is.
In some types of embedded systems, the application software and the operating system software may be indistinguishable to the user, as in the case of software used to control a VCR DVD player or Microwave oven.

Typical examples of software applications are desktop, word processors. spreadsheets and media players.

II READING

1. Read and translate the text into Ukrainian.

With the proliferation of microcomputers at home and work, and the increased cost of developing application software, the use of commercially developed software packages has increased. The availability of these application programs provides both businesses and home users an alternative way to acquire application software.

A word processor is an application program that permits entering, manipulating, formatting, and printing, saving, and recovering text. Word processing is the process of manipulating text. The hardware and software used for the job of word processing are called a word-processing system. There are two configurations of word processing systems. The first, used primarily in business, is a dedicated word-processing system, where the hardware and software are designed only for word processing. The second configuration combines software with a multipurpose computer.

Word processing involves a number of activities, but the activities fall into the category of either text editing or print formatting. In text editing, the user enters text into the system where changes can be made and then saved. Print formatting is when the word processor communicates with the printer through the computer to specify document printing.

Data managers are used to computerize the recording and filing of information. Manual filing systems that consist of file cabinets and folders can be replaced by data managers. Data managers allow the user to add/delete information, search a file based on some criterion, update information, and sort information into some order, and print labels or reports.

There are two kinds of data managers, file handlers and databases. File handlers were developed first and mimic manual methods of filing. File handlers access only one data file at a time. They also cause duplication of data between files when used where many files containing similar information must be maintained. Databases consolidate various independent files into one integrated file from which users draw needed information. The way in which data is stored and organized differentiates a database from a file handler. Databases eliminate data duplication found in file handlers.

Modeling software is based on a mathematical model. A model is a real-world situation represented mathematically. The most popular use of modeling software is in decision-making. By developing a model with variables and then changing these variables, the decision maker can see how various changes affect the model. This process is called simulation.

Electronic spreadsheets are used with microcomputers for modeling. Composed of a table of rows and columns, it is used to manipulate numeric data. A cell is the intersection of a row and column and contains labels, values, and formulas.

To display images such as a pie, bar, or line chart, a graphics software package is necessary. This software allows the user to display images on the screen or print out hard copy. Operating a graphics package may involve selecting choices from a menu. It may also involve controlling individual dots, or pixels, on the screen.

Graphics programs are used in business to prepare pie, bar, or line charts that are then used in presentations. Technically oriented firms use graphics software in computer-aided design. Artists also use graphics software. Some graphics packages permit three-dimensional displays, but most are only two-dimensional. Integrated software consists of normally separate application programs that provide easy movement of data across applications and use a common group of commands. Horizontal integration involves combining application programs into one package that can share data. Vertical integration means an enhancement is made to a single package.

Windows are an enhancement to the operating system that allows several applications to run at the same time. Windows are found primarily on microcomputers.

2. Make up the logical scheme of the text and render the content of the text on the basis of the scheme.

3. Fill in the gaps with the words from the list.

a. mouse	g. designated
b. undo	h. graphics
c. cell	i. data managers
d. word wrap	j. animation
e. system	k. simulation
f. dedicated	1. application

1. Usually, a(n) _____word-processing system is used in a business when large quantities of word processing must be done.

2. If a word is too long to fit correctly on a line, the word processor uses a feature called _____ to position the word on the next line.

3. If you accidentally delete text in a word processor, you may use the _____ option to retrieve the lost text.

4. The ability to search a file based on some standard is a basic feature of _____.

5. When a supervisor uses an electronic spreadsheet to estimate sales growth based on various factors, this is an example of _____.

6. The point in an electronic spreadsheet where a row and column intersect is called a(n) _____.

7. In computer-aided design, engineers can use_____ packages to design products ranging from farm tractors to sophisticated nuclear submarines.

8. The ability to move an image around the display screen is called _____and is important in the preparation of video games.

9. One standard of software integration is that the software is composed of distinct _____programs.

10. An advantage of using a window environment is the use of a(n) _____ as a pointing device.

4. Translate the sentences into the Ukrainian language. Take into consideration the information presented in the sentences. It may be helpful to you while doing the next exercises and discussing the topic.

1. A page-oriented word processor acts as though the document is a series of pages; the user can view and edit only one page at a time. 2. Boiler plating, often used to personalize form letters, is when the same block of text is used in many documents. 3. Windows are used to split the display screen into several autonomous displays that allow different parts of the electronic spreadsheet to be viewed at the same time. 4. Horizontal integration is the combining of several application packages-such as a data manager, electronic spreadsheet, and word processor. 5. Some of the applications for which file handlers have been used include keeping an inventory of personal items, keeping track of Christmas card lists, and creating an appointment calendar. 6. A graphics software package allows a user to enter data and display a pie chart, for example, to show relationships between data. 7. An electronic spreadsheet is a table for rows and columns used to store and manipulate numeric data. 8. Modeling packages, which are used on minicomputers, microcomputers, and mainframe computers, use the computer's speed and power to perform mathematical calculations. 9. The cursor, which usually appears as a blinking line or solid box, indicates where the next typed character will appear. 10. Variable column width is a feature found on many electronic spreadsheets, not data managers.

5. Choose the right answer.

1. When a user enters text via computer into a word processor, it is an example of _____.
   a. text editing
   b. keyboard interfacing logic
   c. keyboard exercise
   d. artificial intelligence

2. A word processor that uses a _____ will give the user a picture on the screen of how the document will appear when it is printed.
   a. file editor
   b. line editor
   c. screen editor
   d. paragraph mistake editor

3. A word processor that eliminates the need to work on pages separately is called a _____word processor.
   a. page-oriented
   b. pseudo-page
   c. document-oriented
   d. text-handler

4. The _____ indicates the current location on the screen and where the next typed character will appear.
   a. status area
   b. cursor
   c. last typed entry
   d. editor

5. Of the two types of data managers, _____ were developed first.
   a. file handlers
   b. data bases
   c. filing systems
   d. data encryption files

6. _____ have become popular for home use to organize and keep records of information.
   a. Word processors
   b. Modems
   c. Databases
   d. File handlers

7. Which is not a standard feature of a data manager?
   a. Add/delete
   b. Sort
   c. Variable column width
   d. Search/update

8. A _____ uses the computer's speed to perform mathematical calculations.
   a. calculator
   b. modeling package
   c. dedicated word processor
   d. model

9. When a user develops a model of a real-world situation and changes variables to see how they influence the model, the process is called _____.
   a. simulation
   b. modeling forecast
   c. if-when analysis
   d. estimate regression

10. The _____ is used with microcomputers for modeling purposes.
   a. general ledger sheet
   c. data base

b. t-account sheet d. electronic spreadsheet

11. A package that allows a user to enter data and then receive a pictorial representation of that data is called a(n) _____ software package.
a. simulation c. education
b. ledger d. graphics

12. Graphics packages can use all of the following except _____ to position the cursor.
a. joysticks c. light pens
b. plotters d. control characters

13. High-resolution graphics differ from low-resolution graphics because the number of _____ available on the display screen is greater.
a. cursor positions c. windows
b. characters d. pixels

14. Adding a spelling program to a word processor is an example of _____.
a. updating c. horizontal integration
b. vertical integration d. update integration

15. _____ allow the user to split the screen into two or more miniscreens displaying application packages that are currently in use or are capable of being used.
a. Switchers c. File handlers
b. Vertically integrated software d. Windows

6. Translate the words and word-combinations into English.

База даних, табличний процесор, редактор рядків, інтеграція, форматування зображення на екрані, відміна (команди), шаблон, автоматичний перехід до нового рядка, графічний пакет програм, курсор, комірка пам'яті, програма опрацювання даних, інтегрована система, елемент зображення, екранний редактор, прокручування, редагування тексту, вікно, вставка, вилучення.

7. Say true or false and explain.

1. T F For large organizations, the cost of developing and maintaining application programs is substantial.
2. T F A word processor is the process of manipulating text using a computer.
3. T F Using a line editor allows a user to edit all the text shown on the entire screen.
4. T F A page-oriented word processor allows only one page of text to reside in internal memory at any one time.

5. T F The primary difference between file handlers and data-base packages is the method of storing and retrieving data.

6. T F A data base consolidates several independent files into one large file that can be accessed by all users in an organization.

7. T F A template is a predefined set of formulas that can be used with a graphics software package.

8. T F Automatic spillover is a feature of some spreadsheets that automatically adjusts the column widths on the spreadsheet when large values are required.

9. T F Some graphics packages have the ability to display four-dimensional images.

10. T F A window environment allows a user to transfer or move data among applications.

III DISCUSSING

1. Give the short answer.

1. How may word processors treat text files differently?
2. List four features usually available on word processors with print-formatting capabilities.
3. What is the difference between a data base package and a file handler?
4. Explain scrolling and differentiate between vertical and horizontal scrolling.
5. What is boiler plating?
6. Why might an organization want to use a database?
7. How are windows used in an electronic spreadsheet?
8. How are titles used in an electronic spreadsheet?
9. What is horizontal integration?
10. List four advantages that windowing software has over conventional application packages.

2. Prepare reports using Internet or other sources (catalogues, magazines, books, etc.) about the latest news, achievements in the field concerning the topic of the chapter.

3. Summarize everything you have learnt on the topic.

4. Answer the question, taking into consideration quick pace of the latest achievements and developments in computer engineering and programming and information presented in the reports, what information from the texts of the Chapter should be reconsidered, changed, added, etc...

5. It's interesting to know…

Word Processor

A word processor (also more formally known as a **document preparation system)** is a computer application used for the production (including composition, editing, formatting, and possibly printing) of any sort of viewable or printed material.

They are descended from early **text formatting** tools (sometimes called **text justification** tools, from their only real capability). Word processing was one of the earliest applications for the personal computer in office productivity.

Although early word processors used tag-based markup for document formatting, most modern word processors take advantage of a graphical user interface. Most are powerful systems consisting of one or more programs, which can produce any arbitrary combination of images, graphics and text, the latter handled with full-blown type-setting capability.

Characteristics

The 'word processing' typically refers to text manipulation functions such as automatic generation of

- batch mailing using a form letter template and an address database
- index of keywords and their page numbers,
- table of contents with section titles and their page numbers,
- table of figures with caption titles and their page numbers,
- 'see also' cross-referencing with page numbers.

Page number and footnote information is extremely hard to maintain without a word processor because addition or deleting of text can affect pagination i.e. page numbers can change in each edition. Other word processing functions include spelling and grammar checking.

Word processors can be distinguished from several other, related forms of software:

Text editor programs were the precursors of word processors. While offering facilities for composing and editing text, they do not offer direct support for document formatting, but batch document processing systems such as LaTeX and programs that implement the paged-media extensions to HTML and CSS fill this gap. Text editors are now used mainly by programmers and web site designers for creating and modifying computer programs, and by computer system administrators for creating and editing configuration files.

Desktop publishing programs, meanwhile, were specifically designed to allow elaborate layout for publication, but offer only limited support for editing. Typically, desktop publishing programs allow users to import text that they have written using a text editor or word processor.

The word processor has become a central component of the office applications suite and is increasingly only available in this form, rather than as a standalone program.

Origin of word processing

The term **word processing** was devised by IBM in the 1960s, and originally encompassed all business equipment—including manually operated typewriters— that was concerned with the handling of text, as opposed to data. Electromechanical paper-tape-based equipment such as the Friden Flexowriter had long been available; the Flexowriter allowed for operations such as repetitive typing of form letters (with a pause for the operator to manually type in the variable information). In the sixties, it began to be feasible to apply the technology developed for electronic computers to office automation tasks. IBM's Mag-Card Selectric was an early device of this kind. It allowed editing, simple revision, and repetitive typing, with a one-line display for editing single lines.

In the early 1970s, Lexitron and Vydec introduced pioneering word-processing systems with CRT screen editing, but the real breakthrough occurred in 1976 with the introduction of a CRT-based system by Wang Laboratories. (A Canadian electronics company, Applied Electronic Systems, introduced a similar product in 1974, but went into bankruptcy a year later. In 1976, refinanced by the Canada Development Corporation, it returned to operation as AES Data, and went on to successfully market its brand of word processors worldwide until its demise in the mid-1980s.) This was a true office machine, affordable by organizations such as medium-sized law firms. It was easily learned and operated by secretarial staff.

The Wang word processor displayed text two-dimensionally on a CRT screen, and incorporated virtually every fundamental characteristic of word processors, as we know them today. The phrase "word processor" rapidly came to refer to CRT-based machines similar to Wang's. Numerous machines of this kind emerged, typically marketed by traditional office equipment companies such as IBM, Lanier (marketing AES Data machines, rebadged), CPT, and NBI. These all, of course, were specialized, dedicated, proprietary systems. Cheap general-purpose computers were still the domain of hobbyists.

With the rise of personal computers, software-based word processors running on general-purpose commodity hardware gradually displaced dedicated word processors, and the term came to refer to software rather than hardware. Early word-processing software was ludicrously clumsy in comparison to dedicated word processors; for example, it required users to memorize semi-mnemonic key combinations rather than pressing keys labeled "copy" or "bold." The cost differences were compelling, however, and personal computers and word processing software soon became serious competition for the dedicated machines.

The late 1980s, saw the advent of laser printers, graphic user interfaces (pioneered by the Xerox Alto and *Gypsy* word processor), and a "typographic" approach to word-processing (WYSIWYG displays with multiple fonts). These were popularized by Microsoft Word on the IBM PC in 1983, and Mac Write on the Apple Macintosh in 1984; these were probably the first true WYSIWYG word processors to become known to a large group of users. Dedicated word processors became museum pieces.

Word processing programs

- Microsoft Word — the most popular word processor at present
- WordPerfect — the most popular word processor of the 1990s, still the gold standard for lawyers and other professionals
- Easy Office Premium — Easy Office is the third most popular word processor in the world at present
- AppleWorks — contains word processor unit derived from Mac Write II
- Framework
- Word MARC — A multi-platform, scientific word processor which ran on both minicomputers (such as Prime and Digital VAX) and PCs.
- Lotus Word Pro (originally Ami Pro)
- Star Office Writer — a branded version of Open Office. org marketed by Sun Microsystems
- Adobe Frame Maker — also used for desktop publishing Mellel — multilingual word processor for Mac OS X
- Nisus Writer Express for Mac OS X
- Mariner Write for Mac OS X

Spreadsheet

A spreadsheet is a rectangular table (or grid) of information, often financial information. (It is, therefore, a kind of matrix.) The word came from "spread" in its sense of a newspaper or magazine item (text and/or graphics) that covers two facing pages, extending across the centerfold and treating the two pages as one large one. The compound word "spread-sheet" came to mean the format used to present bookkeeping ledgers — with columns for categories of expenditures across the top, invoices listed down the left margin, and the amount of each payment in the cell where its row and column intersect, for example - which were traditionally a "spread" across facing pages of a bound ledger (book for keeping accounting records) or on oversized sheets of paper ruled into rows and columns in that format and approximately twice as wide as ordinary paper.

One of the first commercial uses of computers was in processing payroll and other financial records, so the programs (and, indeed, the programming languages themselves) were designed to generate reports in the standard "spreadsheet" format bookkeepers and accountants used. The more available and affordable computers themselves became in the last quarter of the 20th century,

the more software became available for them, and programs to keep financial records and generate spreadsheet reports were always in demand. Those spreadsheet programs can be used to tabulate many kinds of information, not just financial records, so the term "spreadsheet" has developed a more general meaning as information (data = facts) presented in a rectangular table, usually generated by a computer.

Educational research supports the use of spreadsheets both in K-12 and teacher education and in professional development. Dudgale reports a project that involved experienced K-12 teachers in mathematical modeling and problem solving using spreadsheets and concludes that teachers developed models that exhibited a wide variety of mathematics topics and approaches in different grade levels.

Spreadsheet programming

Just as the early programming languages were designed to generate spreadsheet printouts, programming techniques themselves have evolved to process tables (spreadsheets = matrices) of data more efficiently in the computer itself. A **spreadsheet program** is designed to perform general computation tasks using spatial relationships rather than time as the primary organizing principle. Many programs designed to perform general computation use timing, the ordering of computational steps, as their primary way to organize a program. A well-defined entry point is used to determine the first instructions, and all other instructions must be reachable from that point.

In a spreadsheet, however, a set of cells is defined, with a spatial relation to one another. In the earliest spreadsheets, these arrangements were a simple two-dimensional grid. Over time, the model has been expanded to include a third dimension, and in some cases a series of named grids. The most advanced examples allow inversion and rotation operations, which can slice, and project the data set in various ways.

The cells are functionally equivalent to variables in a sequential programming model. Cells often have a formula, a set of instructions, which can be used to compute the value of a cell. Formulas can use the contents of other cells or external variables such as the current date and time. It is often convenient to think of a spreadsheet as a mathematical graph, where the nodes are spreadsheet cells, and the edges are references to other cells specified in formulas. This is often called the dependency graph of the spreadsheet. References between cells can take advantage of spatial concepts such as relative position and absolute position, as well as named locations, to make the spreadsheet formulas easier to understand and manage.

Spreadsheets usually attempt to automatically update cells when the cells on which they depend have been changed. The earliest spreadsheets used simple tactics like evaluating cells in a particular order, but modern spreadsheets compute a minimal recomputation order from the dependency graph. Later

spreadsheets also include a limited ability to propagate values in reverse, altering source values so that a particular answer is reached in a certain cell. Since spreadsheet cells formulas are not generally invertible, though, this technique is of somewhat limited value.

Many of the concepts common to sequential programming models have analogues in the spreadsheet world. For example, the sequential model of the indexed loop is usually represented as a table of cells, with similar formulas. Cyclic dependency graphs produce the traditional construct known as the infinite loop. Most spreadsheets allow iterative recalculation in the presence of these cyclic dependencies, which can be either directly controlled by a user or which stop when threshold conditions are reached.

The power of spreadsheets derives largely from the fact that human beings have a well-developed intuition about spaces, and a well-developed notion of dependency between items. Thus, many people find it easier to perform complex calculations in a spreadsheet than writing the equivalent sequential program.

Origins of the Spreadsheet

The concept of an electronic spreadsheet was outlined in the 1961 paper "Budgeting Models and System Simulation" by Richard Mattessich. Some credit for the computerized spreadsheet perhaps belongs to Pardo and Landau, who filed a patent (U. S. Patent no. 4, 398, 249) on some of the related algorithms in 1970. While the patent was originally rejected by the patent office as being a purely mathematical invention, Pardo and Landau won a court case establishing that "something does not cease to become patentable merely because the point of novelty is in an algorithm." This case helped establish the viability of Software patents.

The generally recognized inventor of the spreadsheet is Dan Bricklin. Bricklin has spoken of watching his university professor create a table of calculation results on a blackboard. When the professor found an error, he had to tediously erase and rewrite a number of sequential entries in the table, triggering Bricklin to think that he could replicate the process on a computer, using a blackboard/spreadsheet paradigm to view results of underlying formulas.

His idea became VisiCalc, the first spreadsheet, and the "killer application" that turned the personal computer from a hobby for computer enthusiasts into a business tool.

Text editor

A text editor is a piece of computer software for editing plain text. It is distinguished from a word processor in that it does not manage document formatting or other features commonly used in desktop publishing.

Text editors are often provided with operating systems or software development packages, and can be used to change configuration files and programming language source code.

Some text editors are small and simple, while others offer a broad and complex range of functionality. For example, UNIX and Unix-like operating systems have the vi editor (or a variant), but many also include the Emacs editor to edit text as well. Microsoft Windows systems come with the very simple Notepad, though many people (especially programmers) use a more complete program like Text Pad. Under Apple Macintosh's classic Mac OS, there was the native SimpleText, which was replaced or supplemented by World Text. Under Mac OS X, Text Edit is the included default editor, which saves files in text and other formats. BBEdit is the most popular third party text and source code editor, which is especially popular for creating HTML content. Windows and Mac OS ports of Emacs also exist.

Types of text editors

Text editors geared for professional computer users place no limit on the size of the file being opened. In particular, they start quickly even when editing large files, and can edit files that are too large to fit the computer's main memory. Simpler text editors often just read files in an array in RAM. On larger files, this is slow, and very large files often do not fit.

The ability to read and write very large files is needed by many professional computer users. For example, system administrators may need to read long log files. Programmers may need to change large source code, or examine naturally large texts, such as an entire dictionary placed in a single file.

Some text editors include specialized computer languages to customize the editor. For example, EMACS can be customized by programming in Lisp. These usually permit the editor to simulate the keystroke combinations and features of other editors, so that users don't have to learn the native command combinations.

Many text editors for software developers include source code syntax highlighting and automatic completion to make a programming language easier to read and write. Programming editors often permit one to select the name of a subprogram or variable, and then jump to its definition and back. Often an auxiliary utility is used to locate the definitions.

Some editors include special features and extra functions, for instance,

- Source code editors
- IDEs
- HTML editors
- Outliners are packages with text editors included, usually with extra functionality.

UNIT 5

SYSTEM ANALYSIS AND DESIGN

I PRE-READING

1. Recollect everything you studied concerning the topic, make notes. Write down the list of words in the Ukrainian language you think you will need while speaking on the topic.

2. a). Read the text and try to guess whether there are words from your list in it.

b). If you don't know the words, translate them using the dictionaries.

c). Choose the words from the list which are the most relevant for this topic.

d). If you don't know the meaning of any words or abbreviations, find their explanations in the Glossary.

e). Which words from exercises 1, 2c may be considered as general notions and which ones as technical terms?

f). Give explanations of several terms, using the Glossary.

A system is a group of related elements that work together toward a common goal. A computer system can help an organization meet its information needs. There are five basic stages in the development of an organization's computer system: system analysis, system design, programming, implementation, and audit and review.

In the analysis phase, data is gathered. Interviews, system flowcharts, questionnaires, and formal reports provide internal sources for data. External sources include customers, suppliers, books, and periodicals. Data is then analyzed and a system analysis report is generated for management.

II READING

1. Read and translate the text into Ukrainian.

When a system analysis report is approved by management, the design stage begins. In the design phase, goals and objectives are reviewed, a system model is developed, organizational restraints are evaluated, alternative designs are developed, feasibility and cost/benefit analyses are performed, and a system design report is generated for management. The analyst's recommendation for a system is included in this report.

The system must next be programmed. Each program is developed as a separate module, and testing is performed as each module is completed. When all individual program tests are done, the entire system is tested and debugged. Program and system documentation is also a vital part of system programming.

There are a number of ways to implement a system. When parallel conversion is used, the old and the new system run together for a period of time. With this approach, no data is lost if the system fails. In pilot conversion, the new system is implemented in only one part of the organization, so that problems can be remedied before full implementation. Phased conversion involves gradual replacement of the old system with the new one. Crash conversion takes place all at once.

After implementation, a system must be audited to decide whether the objectives formulated in system analysis have been met and to find any problems that might affect efficient system operation. System maintenance and follow-up involves continued observation of system operations to ensure that appropriate modifications are made, as management needs change.

System security involves setting up procedures to protect the system from fire, natural disasters, environmental problems, and sabotage. These procedures include emphasizing employee awareness, establishing a security force, and implementing a program of screening people who will have access to the system.

Management information systems are designed to provide information to the different levels of management as needed. There are generally three levels of management in an organization: top-level management, involved in strategic decision-making; middle-level management, involved in tactical decision-making; and lower-level management, involved in operational decision-making.

Decision-oriented reporting helps to properly identify the type of report required by managers. Scheduled listings are produced at set intervals and provide routine information. Exception reports are action-oriented and monitor a particular operation's performance. Predictive reports provide models of possible decision outcomes. Demand reports are produced on request and provide specific information whenever it is needed by management.

Structured design provides a way of breaking down problems into logical modules for easier solving. Modules are related, but they can be worked with independently.

There are several common approaches to designing an MIS within the structure of an organization. These include centralized, hierarchical, distributed, and decentralized approaches.

An MIS provides management with information to support structured decisions. A decision support system (DSS) provides information for unstructured decisions. The use of computers with decision support systems is more to speed a manager's decision-making process than to impose particular solutions to problems.

At the heart of a DSS is the decision model. A model is a mathematical representation of an actual system. The model in a particular DSS should be developed by the manager who will use it so that it represents the way he or she views the system. The top management of many firms remains skeptical of scientific management techniques. The future of DSS depends a great deal on whether or not top-level management ultimately reduces their resistance to it.

Microcomputers are becoming a major part of many corporations' management information systems. Some companies use them in a stand-alone mode, while others have them connected to company databases. Corporations are also encouraging employees to use microcomputers by passing along quantity discounts to them.

2. Make up the logical scheme of the text and render the content of the text based on the scheme.

3. Fill in the gaps with the words from the list.

a. models	g. stand-alone
b. inputs	h. management
c. system flowchart	i. pilot
d. hierarchical	j. phased
e. centralized	k. structured
f. menus	1. unstructured

1. The surrounding environment provides a system with _____.
2. The stream of data through a system is illustrated in a(n) _____.
3. Some systems use _____to explain to users available choices or actions.
4. Only a small portion of an organization is converted to the new system in _____conversion.
5. In predictive reports, future result projections are based on decision _____.
6. The most traditional design approach for an MIS is _____ design.
7. A DSS provides information to support _____decisions.
8. Operational decision-making is _____decision making.
9. Possibly the main factor in whether or not DSS is accepted within an organization is _____.
10. Microcomputers were first seen as _____systems for the home or for small business.

4. Translate the sentences into the Ukrainian language. Take into consideration the information presented in the sentences. It may be helpful to you while doing the next exercises and discussing the topic.

1. Since such personal factors as experience and intuition affect the decision-making process, and since quantifying these qualities is impossible for an analyst, the user has to accurately judge them for him- or herself. 2. Crash conversion would be appropriate if an organization's old information system was not functioning at all. 3. Some steps useful in designing an information system are reviewing goals and objectives, developing a system model, evaluating organizational constraints, developing alternative designs, performing feasibility analysis, performing cost/benefit analysis, and preparing a system design report. 4. Developing an information system entails analysis, design, programming, implementation, and audit and review. 5. Today, the need for proper documentation in these older systems is becoming evident since program changes required to meet new information needs are creating problems in related programs. 6. With companies such as IBM, Hewlett-Packard, and Xerox moving into the microcomputer field, top-level management is beginning to investigate and make use of microcomputer technology. 7. Top-level management has a very skeptical attitude toward scientific management techniques, and this slows or even prevents the implementation of DSS in many organizations. 8. Decision support systems tell managers, through a model, what *might* happen, as opposed to management information systems, which show managers what *has happened*. 9. Fire is one of the major physical threats to computer security along with natural disasters, environmental problems, and sabotage. 10. Exception reports monitor the performance of an organization's system, and when results deviate from expectations, a report is generated.

5. Choose the right answer.

1. _____informs the system whether or not preset standards and goals are being met.
   a. Input
   b. Output
   c. Feedback
   d. Process

2. External sources of information used in system analysis are _____.
   a. interviews
   b. product specifications
   c. system flowcharts
   d. formal reports

3. Relationships among components of a system are summarized in _____.
   a. grid charts
   b. system flowcharts
   c. decision logic tables
   d. formal reports

4. Performing cost/benefit analysis is a step in the _____phase of system development.

a. analysis
b. design
c. implementation
d. audit and review

5. System _____helps prevent errors in other parts of the system when changes are made to a program years after the system is installed.

a. debugging
b. testing
c. documentation
d. conversion

6. When _____conversion is used, an old system is gradually replaced by the new system over a period of time.

a. phased
b. crash
c. parallel
d. pilot

7. When the system conversion is completed, the analyst obtains feedback through _____.

a. testing
b. an audit
c. system debugging
d. feasibility analysis

8. A fire in a computer installation can be put out without damaging magnetic storage media and hardware or presenting a hazard for personnel by using fire extinguishers containing _____.

a. water
b. carbon dioxide
c. halon
d. hydrogen

9. Middle-level management is involved in _____decision-making.

a. strategic
b. tactical
c. operational
d. divisional

10. _____reports are action-oriented reports.

a. Scheduled
b. Exception
c. Predictive
d. Demand

11. An MIS design approach that combines the advantages of having independent operating units with the benefits of central coordination and control is _____design.

a. centralized
b. hierarchical
c. distributed
d. decentralized

12. Unstructured decisions are supported by _____.

a. decision management systems
b. management support systems
c. management information systems
d. decision support systems

13. The heart of a DSS is a _____.

a. decision
c. manager

b. model d. result

14. The acceptance of DSS in business is being slowed by resistance from_____.

a. top-level management c. lower-level management
b. middle-level management d. personnel

15. The entry of_____ into the microcomputer market has gained the attention of senior management regarding microcomputer technology.

a. small, independent manufacturers c. communications corporations
b. business equipment companies d. major mainframe vendors

6. Translate the words and word-combinations into English.

Обслуговування системи, автономний режим, структурне проектування, друк, ревізія, система забезпечення прийняття рішень, повідомлення про виняткові ситуації, розподілена система опрацювання даних, інформаційно-керівна система, електронна таблиця, впровадження, координатна сітка, ієрархічний підхід, централізований підхід, перекодування, таблиця розв'язків, системна блок-схема, захист системи, опір, логічний модуль.

7. Say true or false and explain.

1. T F Data is information that has been processed and is useful for decision making.

2. T F Manufacturers' product specifications are a common source of internal information.

3. T F The focus during data analysis is on why certain operations and procedures are being used.

4. T F The structure in decision logic tables is based on the proposition "if this is done, then this condition is met."

5. T F Performing cost/benefit analysis is a step in the design phase of system development.

6. T F Easy-to-use, understandable system designs are said to be user friendly.

7. T F Because of proper documentation, most systems designed in the early 1970s have proven to be easily adaptable to changing management needs.

8. T F Parallel conversion is an appropriate means of system implementation when the old system is not operational.

9. T F Lower-level managers make basically operational decisions.

10. T F The heart of a decision support system is the decision logic table.

III DISCUSSING

1. Give the short answer.

1. What are the steps involved in developing an information system, and briefly, what happens in each step?
2. List and describe two techniques for analyzing data in system analysis.
3. Name at least four steps that are useful in the design phase of system development.
4. Explain why documentation is so important for information system.
5. In a situation where an old information system is inoperable, what is the best conversion method? What are the advantages and disadvantages of this type of conversion?
6. There are a number of physical threats to computer security. Name them and discuss ways of controlling them.
7. Why should users rather than analysts determine the information requirements of a system being designed for an organization?
8. What are four basic structures for an MIS, and what are the characteristics of each?
9. What is a model, and why is it considered the heart of a DSS?
10. Why does top-level management often resist the implementation of decision support systems?

2. Prepare reports using Internet or other sources (catalogues, magazines, books, etc.) about the latest news, achievements in the field concerning the topic of the chapter.

3. Summarize everything you have learnt on the topic.

4. Answer the question, taking into consideration quick pace of the latest achievements and developments in computer engineering and programming and information presented in the reports, what information from the texts of the Chapter should be reconsidered, changed, added, etc..

5. It's interesting to know…

Operating System

The operating system defines our computing experience. It is the first software we see when we turn on the computer, and the last software we see when the computer is turned off. The software enables all the programs we use. The operating system organizes and controls the hardware on our desks and in our hands, yet most users can't say with any certainty precisely what it is that the operating system does.

It is important to realize that not all computers have operating systems. The computer that controls the microwave oven in your kitchen, for example, doesn't need an operating system. It has one set of relatively simple tasks to perform, very simple input and output methods (a keypad and an LCD screen), and simple, never-changing hardware to control. For a computer like this, an operating system would be unnecessary baggage, adding complexity where none is required. Instead, the computer in a microwave oven simply runs a single program all the time. For computer systems that go beyond the complexity of the microwave, however, an operating system can be the key to greater operating efficiency and easier application development. All desktop computers have operating systems. The most common are the Windows family of operating systems, the UNIX family of operating systems and the Macintosh operating systems. There are hundreds of other operating systems available for special-purpose applications, including specializations for mainframes, robotics, and manufacturing, real-time control systems and so on.

At the simplest level, an operating system does two things:

- It manages the hardware and software resources of the computer system. These resources include such things as the processor, memory, disk space, etc.
- It provides a stable, consistent way for applications to deal with the hardware without having to know all the details of the hardware.

The first task, managing the hardware and software resources, is very important, as various programs and input methods compete for the attention of the central processing unit (CPU) and demand memory, storage and input/output (I/O) bandwidth for their own purposes. In this capacity, the operating system plays the role of the good parent, making sure that each application gets the necessary resources while playing nicely with all the other applications, as well as husbanding the limited capacity of the system to the greatest good of all the users and applications.

The second task, providing a consistent application interface, is especially important if there is to be more than one of a particular type of computer using the operating system, or if the hardware making up the computer is ever open to change. A consistent application program interface (API) allows a software developer to write an application on one computer and have a high level of confidence that it will run on another computer of the same type, even if the amount of memory or the quantity of storage is different on the two machines. Even if a particular computer is unique, an operating system can ensure that applications continue to run when hardware upgrades and updates occur, because the operating system and not the application is charged with managing the hardware and the distribution of its resources. Windows 98 is a great

example of the flexibility an operating system provides. Windows 98 runs on hardware from thousands of vendors. It can accommodate thousands of different printers, disk drives and special peripherals in any possible combination.

Within the broad family of operating systems, there are generally four types, categorized based on the types of computers they control and the sort of applications they support. The broad categories are:

- Real-time operating system (RTOS). Real-time operating systems are used to control machinery, scientific instruments and industrial systems. An RTOS typically has very little user-interface capability, and no end-user utilities, since the system will be a "sealed box" when delivered for use. A very important part of an RTOS is managing the resources of the computer so that a particular operation executes in precisely the same amount of time every time it occurs. In a complex machine, having a part move more quickly just because system resources are available may be just as catastrophic as having it not move at all because the system is busy.
- Single-user, single task. As the name implies, this operating system is designed to manage the computer so that one user can effectively do one thing at a time. The Palm OS for Palm handheld computers is a good example of a modern single-user, single-task operating system.
- Single-user, multi-tasking. This is the type of operating system most people use on their desktop and laptop computers today. Windows 98 and the MacOS are both examples of an operating system that will let a single user have several programs in operation at the same time. For example, it is entirely possible for a Windows user to be writing a note in a word processor while downloading a file from the Internet while printing the text of an e-mail message.
- Multi-user. A multi-user operating system allows many different users to take advantage of the computer's resources simultaneously. The operating system must make sure that the requirements of the various users are balanced, and that each of the programs they are using has sufficient and separate resources so that a problem with one user doesn't affect the entire community of users. UNIX, VMS, and mainframe operating systems, such as MVS, are examples of multi-user operating systems.

It's important to differentiate here between multi-user operating systems and single-user operating systems that support networking. Windows 2000 and Novell Netware can each support hundreds or thousands of networked users, but the operating systems themselves are not true multi-user operating systems. The system administrator is the only "user" for Windows 2000 or Netware. The network support and all of the remote user logins the network enables are, in the

overall plan of the operating system, a program being run by the administrative user.

With the different types of operating systems in mind, it is time to look at the basic functions provided by an operating system. The operating system's tasks, in the most general sense, fall into six categories:

- Processor management
- Memory management
- Device management
- Storage management
- Application interface
- User interface

While there are some who argue that an operating system should do more than these six tasks, and some operating-system vendors do build many more utility programs and auxiliary functions into their operating systems, these six tasks define the core of nearly all operating systems. Let's look at the tools the operating system uses to perform each of these functions.

Processor Management

The heart of managing the processor comes down to two related issues:

- Ensuring that each process and application receives enough of the processor's time to function properly
- Using as many processor cycles for real work as is possible

The basic unit of software that the operating system deals with in scheduling the work done by the processor is either a process or a thread, depending on the operating system. It's tempting to think of a process as an application, but that gives an incomplete picture of how processes relate to the operating system and hardware. The application you see (word processor or spreadsheet or game) is, indeed, a process, but that application may cause several other processes to begin, for tasks like communications with other devices or other computers. There are also numerous processes that run without giving you direct evidence that they ever exist. A process, then, is software that performs some action and can be controlled - by a user, by other applications or by the operating system. It is processes, rather than applications, that the operating system controls and schedules for execution by the CPU. In a single-tasking system, the schedule is straightforward. The operating system allows the application to begin running, suspending the execution only long enough to deal with interrupts and user input. Interrupts are special signals sent by hardware or software to the CPU. It's as if some part of the computer suddenly raised its hand to ask for the CPU's attention in a lively meeting. Sometimes the operating system will schedule the

priority of processes so that interrupts are masked, that is, the operating system will ignore the interrupts from some sources so that a particular job can be finished as quickly as possible. There are some interrupts (such as those from error conditions or problems with memory), that are so important that they can't be ignored. These non-maskable interrupts (NMIs) must be dealt with immediately, regardless of the other tasks. While interrupts add some complication to the execution of processes in a single-tasking system, the job of the operating system becomes much more complicated in a multi-tasking system. Now, the operating system must arrange the execution of applications so that you believe that there are several things happening at once. This is complicated because the CPU can only do one thing at a time. In order to give the appearance of lots of things happening at the same time, the operating system has to switch between different processes thousands of times a second. Here is how it happens.

- A process occupies a certain amount of RAM. It also makes use of registers, stacks and queues within the CPU and operating system memory space.
- When two processes are multi-tasking, the operating system allots a certain number of CPU execution cycles to one program.
- After that number of cycles, the operating system makes copies of all the registers, stacks and queues used by the processes, and notes the point at which the process paused in its execution.
- It then loads all the registers, stacks and queues used by the second process and allows it a certain number of CPU cycles.
- When those are complete, it makes copies of all the registers, stacks and queues used by the second program, and loads the first program.

All of the information needed to keep track of a process when switching is kept in a data package called a process control block. The process control block typically contains:

- An ID number that identifies the process
- Pointers to the locations in the program and its data where processing last occurred
- Register contents
- States of various flags and switches
- Pointers to the upper and lower bounds of the memory required for the process
- A list of files opened by the process
- The priority of the process
- The status of all I/O devices needed by the process

When the status of the process changes, from pending to active, for example, or from suspended to running, the information in the process control block must be used like the data in any other program to direct execution of the task-switching portion of the operating system. This process swapping happens without direct user interference, and each process gets enough CPU cycles to accomplish its task in a reasonable amount of time. Trouble can come, though, if the user tries to have too many processes functioning at the same time. The operating system itself requires some CPU cycles to perform the saving and swapping of all the registers, queues and stacks of the application processes. If enough processes are started, and if the operating system has not been carefully designed, the system can begin to use the vast majority of its available CPU cycles to swap between processes rather than run processes. When this happens, it is called thrashing, and it usually requires some sort of direct user intervention to stop processes and bring order back to the system. One way that operating-system designers reduce the chance of thrashing is by reducing the need for new processes to perform various tasks. Some operating systems allow for a "process-life," called a thread, which can deal with all the CPU-intensive work of a normal process, but generally does not deal with the various types of I/O and does not establish structures requiring the extensive process control block of a regular process. A process may start many threads or other processes, but a thread cannot start a process. So far, all the scheduling we have discussed has concerned a single CPU. In a system with two or more CPUs, the operating system must divide the workload among the CPUs, trying to balance the demands of the required processes with the available cycles on the different CPUs. Asymmetric operating systems use one CPU for their own needs and divide application processes among the remaining CPUs. Symmetric operating systems divide themselves among the various CPUs, balancing demand versus CPU availability even when the operating system itself is all that's running. Even if the operating system is the only software with execution needs, the CPU is not the only resource to be scheduled. Memory management is the next crucial step in making sure that all processes run smoothly.

Memory and Storage Management

When an operating system manages the computer’s memory, there are two broad tasks to be accomplished:

- Each process must have enough memory in which to execute, and it neither can run into the memory space of another process nor be run into by another process.
- The different types of memory in the system must be used properly so that each process can run most effectively.

The first task requires the operating system to set up memory boundaries for types of software and for individual applications. As an example, let's look at an imaginary system with 1 megabyte (1,000 kilobytes) of RAM. During the boot process, the operating system of our imaginary computer is designed to go to the top of available memory and then "back up" far enough to meet the needs of the operating system itself. Let's say that the operating system needs 300 kilobytes to run. Now, the operating system goes to the bottom of the pool of RAM and starts building up with the various driver software required to control the hardware subsystems of the computer. In our imaginary computer, the drivers take up 200 kilobytes. Therefore, after getting the operating system completely loaded, there are 500 kilobytes remaining for application processes. When applications begin to be loaded into memory, they are loaded in block sizes determined by the operating system. If the block size is 2 kilobyte, then every process that is loaded will be given a chunk of memory that is a multiple of 2 kilobytes in size. Applications will be loaded in these fixed block sizes, with the blocks starting and ending on boundaries established by words of 4 or 8 bytes. These blocks and boundaries help to ensure that applications won't be loaded on top of one another's space by a poorly calculated bit or two. With that ensured, the larger question is what to do when the 500-kilobyte application space is filled.

In most computers, it is possible to add memory beyond the original capacity. For example, you might expand RAM from 1 to 2 megabytes. This works fine, but tends to be relatively expensive. It also ignores a fundamental fact of computing, most of the information that an application stores in memory is not being used at any given moment. A processor can only access memory one location at a time, so the vast majority of RAM is unused at any moment. Since disk space is cheap compared to RAM, then moving information in RAM to hard disk can greatly expand RAM space at no cost. This technique is called virtual memory management.

Disk storage is only one of the memory types that must be managed by the operating system, and is the slowest. Ranked in order of speed, the types of memory in a computer system are:

- High-speed cache - This is fast, relatively small amounts of memory that are available to the CPU through the fastest connections. Cache controllers predict which pieces of data the CPU will need next and pull it from main memory into high-speed cache to speed up system performance.
- Main memory - This is the RAM that you see measured in megabytes when you buy a computer.

- Secondary memory - This is most often some sort of rotating magnetic storage that keeps applications and data available to be used, and serves as virtual RAM under the control of the operating system.

The operating system must balance the needs of the various processes with the availability of the different types of memory, moving data in blocks (called pages) between available memories as the schedule of processes dictates.

Device Management

The path between the operating system and virtually all hardware not on the computer's motherboard goes through a special program called a driver. Much of a driver's function is to be the translator between the electrical signals of the hardware subsystems and the high-level programming languages of the operating system and application programs. Drivers take data that the operating system has defined as a file and translate them into streams of bits placed in specific locations on storage devices, or a series of laser pulses in a printer. Because there are such wide differences in the hardware controlled through drivers, there are differences in the way that the driver programs function, but most are run when the device is required, and function much the same as any other process. The operating system will frequently assign high-priority blocks to drivers so that the hardware resource can be released and readied for further use as quickly as possible. One reason that drivers are separate from the operating system is so that new functions can be added to the driver - and thus to the hardware subsystems -- without requiring the operating system itself to be modified, recompiled and redistributed. Through the development of new hardware device drivers, development often performed or paid for by the manufacturer of the subsystems rather than the publisher of the operating system, input/output capabilities of the overall system can be greatly enhanced.

Managing input and output is largely a matter of managing queues and buffers, special storage facilities that take a stream of bits from a device, perhaps a keyboard or a serial port, hold those bits, and release them to the CPU at a rate slow enough for the CPU to cope with. This function is especially important when a number of processes are running and taking up processor time. The operating system will instruct a buffer to continue taking input from the device, but to stop sending data to the CPU while the process using the input is suspended. Then, when the process needing input is made active once again, the operating system will command the buffer to send data. This process allows a keyboard or a modem to deal with external users or computers at a high speed even though there are times when the CPU can't use input from those sources. Managing all the resources of the computer system is a large part of the operating system's function and, in the case of real-time operating systems, may be virtually all the functionality required. For other operating systems, though,

providing a relatively simple, consistent way for applications and humans to use the power of the hardware is a crucial part of their reason for existing.

Kinds of Operating Systems

Operating systems can be grouped according to functionality: operating systems for supercomputing, render farms, mainframes, servers, workstations, desktops, handheld devices, real time systems, or embedded systems.

Supercomputing is primarily scientific computing, usually modeling real systems in nature. Render farms are collections of computers that work together to render animations and special effects. Work that previously required supercomputers can be done with the equivalent of a render farm.

Mainframes used to be the primary form of computer. Mainframes are large centralized computers. At one time they provided the bulk of business computing through time sharing. Mainframes and mainframe replacements (powerful computers or clusters of computers) are still useful for some large scale tasks, such as centralized billing systems, inventory systems, database operations, etc. When mainframes were in widespread use, there was also a class of computers known as minicomputers which were smaller, less expensive versions of mainframes for businesses that couldn't afford true mainframes.

Servers are computers or groups of computers used for internet serving, intranet serving, print serving, file serving, and/or application serving. Servers are also sometimes used as mainframe replacements.

Desktop operating systems are used for personal computers.

Workstations are more powerful versions of personal computers. Often only one person uses a particular workstation (like desktops) and workstations often run a more powerful version of a desktop operating system, but workstations run on more powerful hardware and often have software associated with larger computer systems.

Handheld operating systems are much smaller and less capable than desktop operating systems, so that they can fit into the limited memory of handheld devices.

Real time operating systems (RTOS) are specifically designed to respond to events that happen in real time. This can include computer systems that ran factory floors, computer systems for emergency room or intensive care unit equipment (or even the entire ICU), computer systems for air traffic control, or embedded systems. RTOSs are grouped according to the response time that is acceptable (seconds, milliseconds, microseconds) and according to whether or not they involve systems where failure can result in loss of life.

Embedded systems are combinations of processors and special software that are inside of another device, such as the electronic ignition system on cars.

UNIT 6

COMPUTERS IN OUR FUTURE

I PRE-READING

1. Recollect everything you studied concerning the topic, make notes. Write down the list of words in the Ukrainian language you think you will need while speaking on the topic.

2. a). Read the text and try to guess whether there are words from your list in it.

b). If you don't know the words, translate them using the dictionaries.

c). Choose the words from the list which are the most relevant for this topic.

d). If you don't know the meaning of any words or abbreviations, find their explanations in the Glossary.

e). Which words from exercises 1, 2c may be considered as general notions and which ones as technical terms?

f). Give explanations of several terms, using the Glossary.

In 1965 semiconductor pioneer Gordon Moore predicted that the number of transistors contained on a computer chip would double every year. This is now known as Moore's Law, and it has proven to be somewhat accurate. The number of transistors and the computational speed of microprocessors currently doubles approximately every 18 months. Components continue to shrink in size and are becoming faster, cheaper, and more versatile.

With their increasing power and versatility, computers simplify day-to-day life. Unfortunately, as computer use becomes more widespread, so do the opportunities for misuse. Computer hackers—people who illegally gain access to computer systems – often violate privacy and can tamper with or destroy records. Programs called viruses or worms can replicate and spread from computer to computer, erasing information or causing computer malfunctions. Other individuals have used computers to electronically embezzle funds and alter credit histories. New ethical issues also have arisen, such as how to regulate material on the Internet and the World Wide Web. Individuals, companies, and governments are working to solve these problems by developing better computer security and enacting regulatory legislation.

Computers will become more advanced and they will become easier to use. Reliable speech recognition will make the operation of a computer easier. Virtual reality, the technology of interacting with a computer using all of the human senses, will also contribute to better human and computer interfaces.

Standards for virtual-reality program languages, called Virtual Reality Modeling language (VRML), currently are being developed for the World Wide Web.

Breakthroughs occurred in the area of quantum computing in the late 1990s. Quantum computers under development use components of a chloroform molecule (a combination of chlorine and hydrogen atoms) and a variation of a medical procedure called magnetic resonance imaging (MRI) to compute at a molecular level. Scientists used a branch of physics called quantum mechanics, which describes the activity of subatomic particles (particles that make up atoms), as the basis for quantum computing. Quantum computers may one day be thousands to millions of times faster than current computers, because they take advantage of the laws that govern the behavior of subatomic particles. These laws allow quantum computers to examine all possible answers to a query at one time. Future uses of quantum computers could include code breaking and large database queries.

Communications between computer users and networks will benefit from new technologies such as broadband communication systems that can carry significantly more data and carry it faster, to and from the vast interconnected databases that continue to grow in number and type.

II READING

1. Read and translate the text into Ukrainian.

Increasing the speed and power of computers involves packing more and more transistors into smaller spaces, thus reducing the distance that electrical current has to travel. Heat generated by the dense circuitry, the increased number of I/O terminals needed, the occurrence of crosstalk, and the presence of dust particles are problems that plague the development of such dense circuits.

Gallium arsenide is a substance being used to make faster and more efficient chips. They need less power and generate less heat than silicon chips. Gallium arsenide chips are currently being used in military applications since they are resistant to radiation.

Another approach to increasing computer power is the biochip, which will be grown, if the technology is developed, from living material. Biochips could be used as chemical detectors or possibly even as eyes for the blind. They would be able to repair and reproduce themselves.

Parallel processing is a technique that allows faster computing speeds without many of the problems associated with densely packed circuits. Chips are linked to facilitate simultaneous processing.

Lasers can be combined with computers to perform a variety of tasks. Interactive video uses computer-generated text, sound, and graphics to provide the user with a learning aid or with entertainment. Lasers are also used in wafer integration development.

Fiber optics technology is providing improvements in data communication. Data transmission by fiber optics is faster, more accurate, and cheaper than transmission by standard copper cables. The fibers are immune to noise and electromagnetic interference and are difficult to tap.

Today, artificial intelligence is limited to expert systems. Expert systems draw conclusions and make recommendations based on information obtained from large databases. In the future, artificial intelligence may be used to provide vision and a sense of touch for robots. Natural language communication with computers may be facilitated by AI research. Speech recognition systems can also be improved by principles of AI; the study of spectrograms may soon allow voice recognition systems to respond to anyone's voice. Natural language communication and voice recognition systems will make interfacing with the computer much easier, particularly for the handicapped.

The new age of computers has created an information society with many benefits for its members. There are also a number of concerns. Many people are worried that the plastic cards (EFT cards, credit cards, and so on) that are playing increasingly important roles in their lives could be used to monitor their behavior since records are generally kept of all transactions involving the cards.

Databases also present issues other than crime and privacy for members of the information society. Issues include the future of printed reading materials, copyrights and commercial databases, competition between commercial databases and databases marketed on optical disks, transborder data flow, and accuracy of data. There is also speculation that the advent of the information society is bringing with it a new class of disabled people: people unable, due to illiteracy or limited access to computers, to use computer technology.

The combination of microcomputers and robots is continuing to provide a means of fun and learning for children and adults. Moreover, robot arms controlled by microcomputers are already being used to aid the physically disabled.

2. Make up the logical scheme of the text and render the content of the text based on the scheme.

3. Fill in the gaps with the words from the list.

a. gallium arsenide g. vision

b. parallel processing
c. printing
d. ultraviolet
e. dust particles
f. shell
h. fiber optic
i. laser beams
j. silicon
k. simulations
1. transborder data flow

1. Circuits can be ruined by _____.

2. Silicon may soon be replaced by _____as the principle material for constructing chips.

3. A technology that will increase the ability of computers to imitate human thinking patterns is _____.

4. Wafer integration may become more feasible by using _____to correct defects.

5. One expert system, the Knowledge Engineering Environment, provides a _____on which to build a tailor-made expert system.

6. In order to be helpful at home and in hospitals, robots need to be improved in at least one area. This area is _____.

7. Data transmission becomes faster and more accurate over _____cables.

8. The future of _____ seems to be at stake in the information society.
9. Computers make the problem of _____more difficult to control and detect.
10. Educational software is changing to offer more _____.

4. Translate the sentences into the Ukrainian language. Take into consideration the information presented in the sentences. It may be helpful to you while doing the next exercises and discussing the topic.

1. The densely packed chips of ultra large-scale integration create enough heat and use enough power to burn out the chips. Improved methods of etching the chips and liquid-coolant baths to house the circuitry help reduce heat generation. 2. Vision and touch systems for robots, improved speech recognition systems, summarizing abilities, and natural language communication with computers are all possible applications of AI. 3. People are afraid that since all transactions in which the card is used (and if currency is replaced by cards, all transactions will require the use of a card) are recorded, the government will be able to keep track of their actions. 4. If children use educational software that offers simulations, "how-to's" and concept skills, they will gain knowledge that can be used as a frame of reference for further learning. 5. Robot eyes "see" objects like a television camera and convert these images into digital code (Is and Os) signals that are sent to the robot's "brain" or CPU. 6. Gallium arsenide offers less resistance to the flow of electrical current across the chip than silicon; and gallium arsenide chips generate less heat and require lower voltages than silicon chips, and they resist

radiation. 7. Expert systems are designed to draw conclusions and make recommendations based on a large database of information. 8. The fibers in fiber optics are made of glass, which, in turn, is made from sand making the technology very affordable. 9. Interactive video uses graphics and sound along with computer-generated text to present students with video and audio of historical events, visual representations of concepts, and so on. 10. Laser beams act like blowtorches to make the corrections right on the wafer rather than testing and throwing out defective chips.

5. Choose the right answer.

1. Vacuum tubes were responsible for overheating in early computers; in today's computers, the responsibility lies with _____.
   a. the optical fibers
   b. very densely packed circuits
   c. gallium arsenide
   d. the von Neumann bottleneck

2. _____ chips are five to seven times faster than standard chips.
   a. Gold
   b. Silicon
   c. Gallium arsenide
   d. Optic fiber

3. _____ will be made of living material.
   a. Biochips
   b. Gallium arsenide chips
   c. Parallel processing chips
   d. Genetic chips

4. A type of processing that will make computers better able to process data in imitation of human thinking is _____.
   a. serial processing
   b. the von Neumann bottleneck
   c. cross talk
   d. parallel processing

5. Combining _____and computer programming has created a teaching tool known as interactive video.
   a. laser-read optical disks
   b. video cassette recorders
   c. video games
   d. laser graphics

6. Wafer integration may again become a viable design for chips because _____.
   a. the defective chips can be more easily identified
   b. it uses parallel processing
   c. laser beams can be used to correct defects
   d. it can efficiently be combined with optical disk storage

7. The raw material used to make the fibers in fiber optics is _____.
   a. petroleum
   b. sand
   c. copper
   d. gallium arsenide

8. Expert systems _____.
   a. are like enhanced data bases

b. imitate human thought
c. have common sense
d. are the hardware designed to run artificial intelligence programs

9. Research in voice recognition focuses on all of the following except the ability to _____.
a. read
b. accept larger vocabularies
c. recognize different voices
d. accept continuous speech

10. _____show(s) common features for each sound no matter who is speaking.
a. Voicegrams
b. Speech wedges
c. Voice simulation
d. Spectrograms

11. Today, a robot's eyes are best compared to _____.
a. human eyes
b. television cameras
c. the eyes of an insect
d. single-reflex cameras

12. A particular trait of robots that can be used in the home or hospital is the ability to _____.
a be a bin-picker
b. see in two dimensions
c. vacuum
d. recognize natural language commands

13. _____could possibly be used to ensure our "correct" behavior in the near future.
a. Surveillance cameras
b. Cards
c. Two-way video terminals
d. Biochips

14. The sale of databases on _____could hurt online data services financially.
a. optical disks
b. floppy disks
c. hard disks
d. high density tape

15. Computer software that encourages _____ may radically change how children are taught in our school systems.
a. learning by rote
b. learning through memorization
c. independent and creative thinking
d. mathophobia

6. Translate the words and word-combinations into English.

Штучний інтелект, електричний струм, транзистор, метод, лазер, експертна система, паралельне виконання, потік даних, мікросхема, система розпізнавання мовлення, програмування, захист комп'ютерних даних, зворотний зв'язок, модель прийняття рішень, система опрацювання текстів, перемикач, сегментація, відновлювати, мова високого рівня.

7. Say true or false and explain.

1. T F Chip research and manufacturing occurs in clean rooms because cross talk is less likely to occur there.

3. T F Gallium arsenide chips are five to seven times faster than silicon chips, but they generate more heat.

4. T F Experiments with a prototype of the biochip have proven it to have 10 million times the power of today's supercomputers.

5. T F The von Neumann bottleneck refers to problems experienced with serial processing.

6. T F Parallel processing is a type of processing in which various segments of a program are executed simultaneously.

7. T F Some educators believe that interactive video will someday replace the computer, the instructional film, and perhaps even textbooks.

8. T F A problem with fiber optics technology is that the raw materials required to produce the fibers are expensive.

9. T F Expert systems provide computers with common sense.

10. T F The government could possibly control citizens through the use of plastic computer cards the size of credit cards.

III DISCUSSING

1. Give the short answer.

1. Discuss the significance of packing more and more electronic components on a chip.
2. Why is gallium arsenide particularly useful for military applications?
3. What is the von Neumann bottleneck, and what technique has been developed to handle it?
4. Discuss the potential of interactive video.
5. What are some possible future uses of computers with artificial intelligence?
6. Why are expert systems often not considered true artificial intelligence?
7. Describe the "sight" process in robots. What technology may someday improve the sight of robots?
8. What are spectrograms, and why are they important in the development of voice recognition systems?
9. Why are many people worried about the use of cards (EFT cards, credit cards, etc.)?
10. In the information society created by computer technology, who will be the "newly disadvantaged? Why?

2. Prepare reports using Internet or other sources (catalogues, magazines, books, etc.) about the latest news, achievements in the field concerning the topic of the chapter.

3. Summarize everything you have learnt on the topic.

4. Answer the question, taking into consideration quick pace of the latest achievements and developments in computer engineering and programming and information presented in the reports, what information from the texts of the Chapter should be reconsidered, changed, added, etc…

5. It's interesting to know…

How Computer Viruses Works

Computer viruses are mysterious and grab our attention. On the one hand, viruses show us how vulnerable we are. A properly engineered virus can have an amazing effect on the worldwide Internet. On the other hand, they show how sophisticated and interconnected human beings have become. For example, the things making big news right now are the MSBlaster worm and the SoBig virus. The Melissa virus, which became a global phenomenon in March 1999, was so powerful that it forced Microsoft and a number of other very large companies to completely turn off their e-mail systems until the virus could be contained. The ILOVEYOU virus in 2000 had a similarly devastating effect. That's pretty impressive when you consider that the Melissa and ILOVEYOU viruses are incredibly simple. We will discuss viruses, both "traditional" viruses and the newer e-mail viruses, so that you can learn how they work and understand how to protect yourself. Viruses in general are on the wane, but occasionally a person finds a new way to create one, and that's when they make the news.

Types of Infection

When you listen to the news, you hear about many different forms of electronic infection. The most common are:

- Viruses. A virus is a small piece of software that piggybacks on real programs. For example, a virus might attach itself to a program such as a spreadsheet program. Each time the spreadsheet program runs, the virus runs, too, and it has the chance to reproduce (by attaching to other programs) or wreak havoc.
- E-mail viruses. An e-mail virus moves around in e-mail messages, and usually replicates itself by automatically mailing itself to dozens of people in the victim's e-mail address book.
- Worms. A worm is a small piece of software that uses computer networks and security holes to replicate itself. A copy of the worm scans the

network for another machine that has a specific security hole. It copies itself to the new machine using the security hole, and then starts replicating from there, as well.

- Trojan horses. A Trojan horse is simply a computer program. The program claims to do one thing (it may claim to be a game) but instead does damage when you run it (it may erase your hard disk). Trojan horses have no way to replicate automatically.

Computer viruses are called viruses because they share some of the traits of biological viruses. A computer virus passes from computer to computer like a biological virus passes from person to person.

There are similarities at a deeper level, as well. A biological virus is not a living thing. A virus is a fragment of DNA inside a protective jacket. Unlike a cell, a virus has no way to do anything or to reproduce by itself - it is not alive. Instead, a biological virus must inject its DNA into a cell. The viral DNA then uses the cell's existing machinery to reproduce itself. In some cases, the cell fills with new viral particles until it bursts, releasing the virus. In other cases, the new virus particles bud off the cell one at a time, and the cell remains alive.

A computer virus shares some of these traits. A computer virus must piggyback on top of some other program or document in order to be executed. Once it is running, it is then able to infect other programs or documents. Obviously, the analogy between computer and biological viruses stretches things a bit, but there are enough similarities that the name sticks.

What's a "Worm"?

A worm is a computer program that has the ability to copy itself from machine to machine. Worms normally move around and infect other machines through computer networks. Using a network, a worm can expand from a single copy incredibly quickly. For example, the Code Red worm replicated itself over 250,000 times in approximately nine hours on July 19, 2001.

A worm usually exploits some sort of security hole in a piece of software or the operating system. For example, the Slammer worm (which caused mayhem in January 2003) exploited a hole in Microsoft's SQL server. This article offers a fascinating look inside Slammer's tiny (376 byte) program.

Worms use up computer time and network bandwidth when they are replicating, and they often have some sort of evil intent. A worm called Code Red made huge headlines in 2001. Experts predicted that this worm could clog the Internet so effectively that things would completely grind to a halt.

The Code Red worm slowed down Internet traffic when it began to replicate itself, but not nearly as badly as predicted. Each copy of the worm scanned the Internet for Windows NT or Windows 2000 servers that do not have the Microsoft security patch installed. Each time it found an unsecured server, the worm copied itself to that server. The new copy then scanned for other servers to infect. Depending on the number of unsecured servers, a worm could conceivably create hundreds of thousands of copies. The Code Red worm was designed to do three things: replicate itself for the first 20 days of each month, replace Web pages on infected servers with a page that declares "Hacked by Chinese", and launch a concerted attack on the White House Web server in an attempt to overwhelm it.

How They Spread

Early viruses were pieces of code attached to a common program like a popular game or a popular word processor. A person might download an infected game from a bulletin board and run it. A virus like this is a small piece of code embedded in a larger, legitimate program. Any virus is designed to run first when the legitimate program gets executed. The virus loads itself into memory and looks around to see if it can find any other programs on the disk. If it can find one, it modifies it to add the virus's code to the unsuspecting program. Then the virus launches the "real program." The user really has no way to know that the virus ever ran. Unfortunately, the virus has now reproduced itself, so two programs are infected. The next time either of those programs gets executed, they infect other programs, and the cycle continues.

If one of the infected programs is given to another person on a floppy disk, or if it is uploaded to a bulletin board, then other programs get infected. This is how the virus spreads. The spreading part is the infection phase of the virus. Viruses wouldn't be so violently despised if all they did was replicate themselves. Unfortunately, most viruses also have some sort of destructive attack phase where they do some damage. Some sort of trigger will activate the attack phase, and the virus will then "do something", anything from printing a silly message on the screen to erasing all of your data. The trigger might be a specific date, or the number of times the virus has been replicated, or something similar.

As virus creators got more sophisticated, they learned new tricks. One important trick was the ability to load viruses into memory so they could keep running in the background as long as the computer remained on. This gave viruses a much more effective way to replicate themselves. Another trick was the ability to infect the boot sector on floppy disks and hard disks. The boot sector is a small program that is the first part of the operating system that the computer loads. The boot sector contains a tiny program that tells the computer

how to load the rest of the operating system. By putting its code in the boot sector, a virus can guarantee it gets executed. It can load itself into memory immediately, and it is able to run whenever the computer is on. Boot sector viruses can infect the boot sector of any floppy disk inserted in the machine, and on college campuses where many people share machines they spread like wildfire.

In general, both executable and boot sector viruses are not very threatening any more. The first reason for the decline has been the huge size of today's programs. Nearly every program you buy today comes on a compact disc. Compact discs cannot be modified, and that makes viral infection of a CD impossible. The programs are so big that the only easy way to move them around is to buy the CD. People certainly can't carry applications around on a floppy disk like they did in the 1980s, when floppies full of programs were traded like baseball cards. Boot sector viruses have also declined because operating systems now protect the boot sector. Both boot sector viruses and executable viruses are still possible, but they are a lot harder now and they don't spread nearly as quickly as they once could. Call it "shrinking habitat," if you want to use a biological analogy. The environment of floppy disks, small programs and weak operating systems made these viruses possible in the 1980s, but that environmental niche has been largely eliminated by huge executables, unchangeable CDs and better operating system safeguards.

E-mail Viruses

The latest thing in the world of computer viruses is the e-mail virus, and the Melissa virus in March 1999 was spectacular. Melissa spread in Microsoft Word documents sent via e-mail, and it worked like this: Someone created the virus as a Word document uploaded to an Internet newsgroup. Anyone who downloaded the document and opened it would trigger the virus. The virus would then send the document (and therefore itself) in an e-mail message to the first 50 people in the person's address book. The e-mail message contained a friendly note that included the person's name, so the recipient would open the document thinking it was harmless. The virus would then create 50 new messages from the recipient's machine. As a result, the Melissa virus was the fastest-spreading virus ever seen! As mentioned earlier, it forced a number of large companies to shut down their e-mail systems.

The ILOVEYOU virus, which appeared on May 4, 2000, was even simpler. It contained a piece of code as an attachment. People who double clicked on the attachment allowed the code to execute. The code sent copies of itself to everyone in the victim's address book and then started corrupting files on the victim's machine. This is as simple as a virus can get. It is really more of a Trojan horse distributed by e-mail than it is a virus. The Melissa virus took

advantage of the programming language built into Microsoft Word called VBA, or Visual Basic for Applications. It is a complete programming language and it can be programmed to do things like modify files and send e-mail messages. It also has a useful but dangerous auto-execute feature. A programmer can insert a program into a document that runs instantly whenever the document is opened. This is how the Melissa virus was programmed. Anyone who opened a document infected with Melissa would immediately activate the virus. It would send the 50 e-mails, and then infect a central file called NORMAL.DOT so that any file saved later would also contain the virus! It created a huge mess.

Microsoft applications have a feature called Macro Virus Protection built into them to prevent this sort of thing. With Macro Virus Protection turned on (the default option is ON), the auto-execute feature is disabled. Therefore, when a document tries to auto-execute viral code, a dialog pops up warning the user. Unfortunately, many people don't know what macros or macro viruses are, and when they see the dialog they ignore it, so the virus runs anyway. Many other people turn off the protection mechanism. So, the Melissa virus spread despite the safeguards in place to prevent it. In the case of the ILOVEYOU virus, the whole thing was human-powered. If a person double-clicked on the program that came as an attachment, then the program ran and did its thing. What fueled this virus was the human willingness to double-click on the executable.

An Ounce of Prevention

You can protect yourself against viruses with a few simple steps:

- If you are truly worried about traditional (as opposed to e-mail) viruses, you should be running a more secure operating system like UNIX. You never hear about viruses on these operating systems because the security features keep viruses (and unwanted human visitors) away from your hard disk.
- If you are using an unsecured operating system, then buying virus protection software is a nice safeguard.
- If you simply avoid programs from unknown sources (like the Internet), and instead stick with commercial software purchased on CDs, you eliminate almost all of the risk from traditional viruses. In addition, you should disable floppy disk booting -- most computers now allow you to do this, and that will eliminate the risk of a boot sector virus coming in from a floppy disk accidentally left in the drive.
- You should make sure that Macro Virus Protection is enabled in all Microsoft applications, and you should NEVER run macros in a document unless you know what they do. There is seldom a good reason to add macros to a document, so avoiding all macros is a great policy.

- In the case of the ILOVEYOU e-mail virus, the only defense is a personal discipline. You should never double-click on an attachment that contains an executable that arrives as an e-mail attachment. Attachments that come in as Word files (.DOC), spreadsheets (.XLS), images (.GIF and .JPG), etc., are data files and they can do no damage (noting the macro virus problem in Word and Excel documents mentioned above). A file with an extension like EXE, COM or VBS is an executable, and an executable can do any sort of damage it wants. Once you run it, you have given it permission to do anything on your machine. The only defense is to never run executables that arrive via e-mail.

By following those simple steps, you can remain virus free.

Origins

People create viruses. A person has to write the code, test it to make sure it spreads properly and then release the virus. A person also designs the virus's attack phase, whether it is a silly message or destruction of a hard disk. So why do people do it?

There are at least three reasons. The first is the same psychology that drives vandals and arsonists. Why would someone want to bust the window on someone else's car, or spray-paint signs on buildings or burn down a beautiful forest? For some people that seems to be a thrill. If that sort of person happens to know computer programming, then he or she may funnel energy into the creation of destructive viruses.

The second reason has to do with the thrill of watching things blow up. Many people have a fascination with things like explosions and car wrecks. When you were growing up, there was probably a kid in your neighborhood who learned how to make gunpowder and then built bigger and bigger bombs until he either got bored or did some serious damage to himself. Creating a virus that spreads quickly is a little like that - it creates a bomb inside a computer, and the more computers that get infected the more "fun" the explosion.

The third reason probably involves bragging rights, or the thrill of doing it. Sort of like Mount Everest. The mountain is there, so someone is compelled to climb it. If you are a certain type of programmer and you see a security hole that could be exploited, you might simply be compelled to exploit the hole yourself before someone else beats you to it. "Sure, I could TELL someone about the hole. But wouldn't it be better to SHOW them the hole?" That sort of logic leads to many viruses.

Of course, most virus creators seem to miss the point that they cause real damage to real people with their creations. Destroying everything on a person's hard disk is real damage. Forcing the people inside a large company to waste thousands of hours cleaning up after a virus is real damage. Even a silly message is real damage because a person then has to waste time getting rid of it. For this reason, the legal system is getting much harsher in punishing the people who create viruses.

History

Traditional computer viruses were first widely seen in the late 1980s, and they came about because of several factors. The first factor was the spread of personal computers (PCs). Prior to the 1980s, home computers were nearly non-existent or they were toys. Real computers were rare, and they were locked away for use by "experts." During the 1980s, real computers started to spread to businesses and homes because of the popularity of the IBM PC (released in 1982) and the Apple Macintosh (released in 1984). By the late 1980s, PCs were widespread in businesses, homes and college campuses.

The second factor was the use of computer bulletin boards. People could dial up a bulletin board with a modem and download programs of all types. Games were extremely popular, and so were simple word processors, spreadsheets, etc. Bulletin boards led to the precursor of the virus known as the Trojan horse. A Trojan horse is a program that sounds really cool when you read about it. Therefore, you download it. When you run the program, however, it does something uncool like erasing your disk. So you think you are getting a neat game but it wipes out your system. Trojan horses only hit a small number of people because they are discovered quickly. Either the bulletin board owner would erase the file from the system or people would send out messages to warn one another.

The third factor that led to the creation of viruses was the floppy disk. In the 1980s, programs were small, and you could fit the operating system, a word processor (plus several other programs) and some documents onto a floppy disk or two. Many computers did not have hard disks, so you would turn on your machine and it would load the operating system and everything else off the floppy disk.

Viruses took advantage of these three facts to create the first self-replicating programs.

SUPPLEMENTARY TEXTS

Artificial Intelligence (AI)

Artificial intelligence is a term that in its broadest sense would indicate the ability of an artifact to perform the same kinds of functions that characterize human thought. The possibility of developing some such artifact has intrigued human beings since ancient times. With the growth of modern science, the search for AI has taken two major directions: psychological and physiological research into the nature of human thought, and the technological development of increasingly sophisticated computing systems.

In the latter sense, the term AI has been applied to computer systems and programs capable of performing tasks more complex than straightforward programming, although still far from the realm of actual thought. The most important fields of research in this area are information processing, pattern recognition, game-playing computers, and applied fields such as medical diagnosis.

Current research in information processing deals with programs that enable a computer to understand written or spoken information and to produce summaries, answer specific questions, or redistribute information to users interested in specific areas of this information. Essential to such programs is the ability of the system to generate grammatically correct sentences and to establish linkages between words, ideas, and associations with other ideas. Research has shown that whereas the logic of language structure-its syntax-submits to programming, the problem of meaning, or semantics, lies far deeper, in the direction of true AI. In medicine, programs have been developed that analyze the disease symptoms, medical history, and laboratory test results of a patient, and then suggest a diagnosis to the physician. The diagnostic program is an example of so-called expert systems-programs designed to perform tasks in specialized areas as a human would. Expert systems take computers a step beyond straightforward programming, being based on a technique called rule-based inference, in which pre-established rule systems are used to process the data. Despite their sophistication, systems still do not approach the complexity of true intelligent thought.

Many scientists remain doubtful that true AI can ever be developed. The operation of the human mind is still little understood, and computer design may remain essentially incapable of analogously duplicating those unknown, complex processes. Various routes are being used in the effort to reach the goal of true AI. One approach is to apply the concept of parallel processing-interlinked and concurrent computer operations. Another is to create networks of experimental computer chips, called silicon neurons, that mimic data-processing functions of brain cells. Using analog technology, the transistors in these chips emulate nerve-cell membranes in order to operate at the speed of neurons.

What is High Level Language?

A high-level language is a problem oriented programming language, whereas a low-level language is machine oriented. In other words, a high-level language is a convenient and simple means of describing the information structures and sequences of actions required to perform a particular task. A high-level language is independent of the architecture of the computer, which supports it. This has two major advantages.

Firstly, the person writing the programs does not have to know anything about the computer on which the program will be run.

Secondly, programs are portable, that is, the same program can (in theory) be run on different types of computer. However, this feature of machine independence is not always achieved in practice. In most cases, programs in high-level languages are shorter than equivalent programs in low-level languages. However, conciseness can be carried too far, to the point where programs become impossible to understand. More important features of a high-level language are its ability to reflect clearly the structure of programs written in it, and its readability.

High-level languages may be broadly classified as general-purpose or special-purpose. General-purpose languages are intended to be equally well suited to business, scientific, engineering or systems software tasks. The commonest general-purpose languages are Algol 68 and PL/1. The language Ada also falls into this category. Because of their broad capabilities, these languages are large and relatively difficult to use.

The commonest categories of special-purpose languages arc commercial, scientific and educational. In the commercial field, COBOL still reigns supreme, while FORTRAN is still the most widely used scientific language. In the computer education field, Basic is widely used in schools, with Logo and Prolog gaining popularity. Pascal is the most popular language at universities. Pascal is a powerful general-purpose language in its own right.

Another way of classifying high-level languages is as procedural and declarative languages. Procedural languages state how a task is to be performed, often breaking programs into procedures, each of which specifies how a particular operation is to be performed.

All the early high-level languages are procedural, with Algol, Pascal and Ada as typical examples.

Declarative programming languages describe the data structures and relationships between data relevant to a particular task, and specify what the objective of the task is. The process by which the task is to be carried out is not stated explicitly in the program. This process is determined by the language translation system. Prolog is an example of a declarative programming language. The defining characteristics of a high-level language are problem-orientation and machine independence.

The first objective of a high-level language is to provide a convenient means of expressing the solution to a problem. There arc two other common ways of doing this mathematics, and natural languages, such as English. Most high-level languages borrow, without much modification, concepts and symbols from mathematics. The problem with natural languages is that in their full richness and complexity, they are quite impossible to use to instruct a computer. Nevertheless, high-level languages use words from natural languages, and allow these words, and mathematical symbols, to be combined according to various rules. These rules create the structure of programs written in the language. The result, in a good high-level language, is a clear structure, not too different from our customary ways of thinking and expressing ourselves.

This discussion leads to the second objective of high-level languages simplicity. Simplicity is achieved by a small set of basic operations, a few clear rules for combining these operations, and, above all, the avoidance of special cases.

The third objective of a high-level language is efficiency. Programs in the language must be able to be translated into machine code very quickly, and the resulting machine code must run efficiently. This objective usually conflicts with the first two. Most high-level languages reflect a compromise between these objectives.

The final objective is readability of programs. Many languages allow for the inclusion of comments or additional 'noise' words, to make programs easier to read. However, a good high-level language should enable programs to be written which are clear to read without additional comments. Regrettably, some high level languages ignore this objective altogether.

Features of High Level Languages

The character set used by a language is the set of all characters, which may be used in programs written in the language. Almost all languages use letters and decimal digits.

Most high-level languages use reserved words. These are words, which have a specific meaning in programs, and may not be used by the programmer for any other purpose. For example, in Pascal, reserved words include read, if... then... else and write. Some Languages permit abbreviations of reserved words. The size and complexity of a language can be measured by the number of reserved words it uses.

Perhaps the most important feature of a high-level language is the way in which programs in it arc structured. The structure of a program is specified by a set of rules, called rules of syntax. Different languages have different ways of expressing these rules. In some, the rules arc written in concise English. Others use syntax diagrams, while others (notably Algol) use a notation originally called Backus-Naur form, now known as BNF.

Much attention has been devoted, in the development and use of high-level languages, to (the way in which programs are split up into blocks or modules, each module doing a specific task. In some languages, notably FORTRAN, these blocks are called subroutines, in others such as Algol and Pascal, these blocks are called procedures or functions. Because of the careful structuring of programs into blocks, which they permit, Algol, Pascal and similar languages are called block-structured languages. Procedures, functions or subroutines are activated via calls from other parts of the program. For example, if a program contains a junction to calculate the square root of a given number, this function is called every time a square root is required in the rest of the am. Most languages permit a procedure or function to call itself, a feature known as recursion. This is an extremely powerful feature for handling such data structures as lists, stacks and trees, and for such tasks as analyzing the structure of arithmetic

An important aspect of high-level languages is the way in which they handle the data items and data structures used in a program.

Broadly speaking, data items fall into two categories: variables, which can change their value during the running of a program, and constants, which keep the same value. In most program languages, variables are given names, or identifiers. In some languages, such as FORTRAN and Basic, constants are referred to by their values, while in others, such as Algol and Pascal, constants are also given identifiers.

Some program languages require that all variables be declared before they are used. Generally, variables are declared by listing them at the start of the procedure or subroutine in which they are to be used. An attempt to use a variable which has not been declared results in an error.

This gives rise to the idea of the scope of a variable. The scope of a variable is the part of a program in which it may be used. Variables which are declared for use in one procedure only are called local variables. Their scope is limited to that procedure. Variables which are declared for use in the whole program are called global variables. Their scope is the whole program. The intention of providing each variable with a scope is to enable a program to be broken up into watertight blocks, or modules. Each block uses only the information it requires. This simplifies the task of designing, writing and testing programs, and limits the effects of errors.

Almost all high-level languages include the notion of data types. In Basic language, the standard data types are numeric and character strings. These types can be incorporated into arrays, which are tables of items of the same type. In most high-level languages, numbers can be integers or real numbers (generally stored in floating point form). PL/1 even permits the number of significant figures in a number to be declared. Another common standard data type is Boolean, with the range of values 'true' and 'false'. Data types can contain single elements, or be structures such as arrays, stacks, lists, trees, etc.

Computer Personnel

There are two overall stages in the manufacture of computers, namely original equipment manufacture, and the design and assembly of complete computer systems.

Integrated circuits are designed and developed by electronics engineers, and fabricated by highly skilled workers using sophisticated equipment. Computers are used in the design, manufacture and testing of these circuits.

The complete process of designing and constructing a computer is extremely complex, and involves the work of a number of people. Most computer manufacturers have a research department, investigating new computer architectures, new hardware devices, new software techniques and new computer applications. Scientists, research engineers and technicians, as well as highly skilled software engineers are among the staff of theses departments.

The overall design of a new computer, or series of computers, is in the hands of computer architects. Modern computers are designed from both the hardware and the software point of view. Accordingly, systems programmers, who write the systems software for the computer, are also involved in the design process.

Highly skilled, production workers are responsible for the various stages of construction and assembly of units. Production lines are not used. Generally, a team of workers is assigned to take a unit through all stages of construction and exhaustive testing.

One of the highest paid jobs in computing is that of computer salesman. Salesmen operate in an intensely competitive environment, where their level of pay depends to some extent on their sales figures. The process of selling a large computer system can take several months.

Field engineers care-responsible for the installation and commissioning of new computer units, and the maintenance and repair of systems in operation. With many computers running 24 hours a day, this type of work often involves calls at unsocial hours.

Traditionally, an Organization, which uses a computer, has a data processing department, containing all the staff who works directly with the computer. Other departments and individuals in an organization relate to the data processing department as users of the computing equipment. The description of a data processing department given here applies in particular to banks, insurance companies, airlines and many central and local government departments.

In overall charge of a data processing department is a data processing manager. Responsibilities of a data processing manager include formulation of policy, approval of projects, staff recruitment and maintaining the relationship of the department with the rest of the company.

The work of a data processing department has two aspects, namely the program development aspect, and the operations aspect.

On the program development side, systems analysts are responsible for steering each project through the data processing cycle. Programmers, more properly called applications programmers in this context, are responsible for writing, correcting and maintaining programs, and producing various items of documentation, explaining how programs are used.

On the operation, side, there is an operations manager in charge of scheduling the use of the computer, arranging for maintenance and ordering supplies. A team of operators man the computer room, often working in shifts under shift leaders. The flow of data to and from the computer is sometimes supervised by data controllers, who ensure that the right data is available at the right time. Data entry staff operates data entry terminals to keep up the supply of data to the computer. In many large computer systems, file librarians are responsible for the large numbers of magnetic tapes and magnetic disks used. If the computer supports a database, then a database administrator is in charge of this aspect of the work. If a data communication network is used, this is often under the overall charge of a network administrator.

In many organizations, computers have become such an integral part of their operations that a separate data processing department is not required. The majority of the workers in such organizations make some use of the computer. Programming and systems design is often contracted out to software houses, or standard software packages are purchased. An expanding computer application in this category is word processing. In many modern offices, all correspondence, contracts and similar work is done on word processing systems. Most of the people working in these offices use a word processing workstation to some extent.

Situated between computer manufacturers and computer users are software houses, or computer service bureaus. These provide a wide range of services, including consultancy on computer projects, designing software to a customer's specifications, selling software and supplying complete computer systems.

Many software houses have extremely flexible working arrangements, falling broadly under the heading of software engineering. Most of their workers are skilled in a variety of areas, including systems and applications programming, systems analysis, project management and computer design.

A team of workers is assigned to each project, which is undertaken. Work is shared among the members of the team according to their interests, capabilities and experience. When a project has been completed, the team is disbanded. Workers arc assigned to other projects.

Protection of Information

Rapid development of automation processes and the penetration of the computers in all fields of life have lead to appearance of a range of peculiar problems. One of these problems is the necessity of providing effective protection to information and means of its processing.

Many ways to access information, considerable quantity of qualified specialists, and vast use of special technical equipment in social production make it possible for violators practically at any moment and in any place carry out the actions, which represent a threat to information safety. Particular role in this process has been played by appearance of personal computer (PC), which has made computers, software and other informational technologies available to general public. Wide distribution of PC and impossibility of conducting effective control of their use have resulted in the decreasing security level of information systems.

In the current situation, data processing has moved the problems of information security forward to the rank of most important problems of national economy. Solving the problem of poor information security presupposes a complex of measures. First, such actions of government as development of classification system, documentation of information and protection methods, data access regulations and punishing measures against information security violators. Formation of state informational sources is carried out by citizens, state authorities, organizations and social unions. Documents, which belong to a person, can be included in the state structure of informational sources, of course, if the person wishes. State informational sources are open and generally available. Documented information with limited access is divided into state secret and confidential information.

Personal data refers to confidential information. The collection, storage, use and distribution of private information are not allowed. The information, which breaks personal and family secret, secret of correspondence, telephone, postal, telegraph talks and other messages of a person without his/her permission, is also confidential. Personal data may not be used with purpose of causing damage to person's property and reputation, difficulties of realization its right. Collected data must be limited to necessary information. The information, which carries strong probability of causing damage to a citizen’s interests shouldn’t be collected.

There are some categories of personal information:

- secret documents;
- official department rules and instructions;
- information, which is not to be made public in accordance with legislative acts;
- confidential business information;
- information, which touches private life of a person;

• information of financial institutions;

All types of informational systems and networks, technologies and means of their providing compose a special branch of economic activity, whose development is defined by the state scientific, technological and industrial policy of informatization. State and non-state organizations and, of course, the citizens have equal rights in terms of access to the development and producing of informational systems, technologies.

The informational systems, technologies and means of their providing can be the property objects of juridical person, non-juridical person and state. The owner of informational system is a person, who purchased these objects or got as a gift, heredity or by any other legal way. The informational systems, technologies and means of their providing can be considered as a good (product), if the producer rights are not broken. The owner of informational system determines the using conditions of this product.

Copyrights and property rights on informational systems, technologies and means of their providing can be belong to different persons. The owner of informational systems has to protect copyrights in accordance with legislation.
Informational systems and databases, intended for citizens' and organizations informational service, are subjected to certification according to the established custom. The organizations, which work in the field of making design, producing the means of information protection and personal data treatment, must obtain licensees to conduct such activity. The steps for obtaining license are defined by the legislation.

Computer Systems and Protection of Information

The problem of information security is relatively new. Not all problems, connected with it have been figured out and solved up to now. The fact of great number of computer systems users means the definite risk to security because not all clients will carry out the requirements of its providing.
The order of storage mediums should be clearly defined in legal acts and envisage the complete safety of mediums, control over the work with information, responsibility for unsanctioned access to mediums with a purpose of copying, changing or destroying them and so on.

There are some legal aspects of information protection, which can appear due to not carefully thought or ill-intentioned use of computer techniques:

• legal questions of protection of informational massifs from distortions;
• security of stored information from the unsanctioned access;
• setting juridical fixed rules and methods of copyrights protection and priorities of software producers;
• development of measures for providing the juridical power to the documents, which are given to the machines;
• legal protection of the experts' interests, who pass their knowledge to the databases;

• setting of legal norms and juridical responsibility for using electronic computer means in personal interests, which hurt other people and social interests and can harm them;

The lack of appropriate registration and control, low level of work and production personnel discipline, the access of an unauthorized persons to the computing sources create conditions for abusing and cause difficulties to their detection. In every computing center, it is usual to set and strictly follow the regulations of the access to different official rooms for employees of any categories.

The main purpose of information protection is preventing from the leak, theft, distortion, counterfeit of information; preventing the threat to person's life and social safety, protection of the constitution and so on. The information is subjected to protection, when it may cause the harm for its owner, user or other person.

Computer Crimes

The development of computer technology and its wide use have lead to appearance and spread of computer crimes. Such situation causes alarm among those organizations and legislative institutions that use computer technologies and, of course, people, who use new informational services at homes.

The term "computer crime" was first used in the early 70s. However, the discussions concerning it are still actual. The top question of these discussions is "What unlawful actions are implied by computer crime". A rank of definitions of the computer crime has been composed. It often refers to crimes directly or indirectly connected to electronic computing machines and which includes a number of illegal acts, committed by means of electronic data processing system or against it. Others consider that computer crime is any action, which goes together with interfering with property rights and fulfilled by means of computers. The thirds think that computer crime can be defined as all intentional and unlawful actions, which lead to causing harm to possessions, with help of computers too.

There are following forms of computer criminality: computer manipulations, economic espionage, sabotage, computer extortion, "hackers" activity. The main character of committing computer crimes in the business field becomes highly qualified "white collars" from the suffered organization's employees.

According to the MIS Trading Institute (USA), they get 63% of all causes, examining crimes and abuses. More than 36% of law-committing employees are related to the personnel, which is not connected with computer servicing, 29% -qualified programmers, 25% - other workers of computing center. This tendency is reflected in official statistics too, according to which, about 40%

of computer crimes are committed for solving of financial problems, 20% are motivated as an intellectual challenge to society, 17% - by the willing of solving personal problems, 8% - problems of corporation or organization, 4% - are directed for social admitting, 3% - for wounding somebody's rights and so on.

The most dangerous individuals of computer swindle are so called "hackers", "crackers" and representatives of other groups, working in the sphere of industrial espionage. So, many security specialists advise employers to pay special attention to engaged workers-specialists in computer technologies, programming and information protection spheres.

There are many causes, when "hackers" get a job with a goal of personal enrichment. But the most danger can represent such specialists, who are in collusion with managers of commercial structures and organized criminal groups; in these situations causing damage and weight of consequences considerably increases.

There are two types of unsanctioned access:

- internal "breaking open" - the criminal has access to the terminal, with information he interested in and can work with it for some time without somebody's control;
- external “breaking open” – the criminal doesn’t have indirect access to the computer system, but has an opportunity of penetration to the protected system by means of remote access.

The History of Computers in the USA

Howard Aiken's contributions to the development of the computer -notably the Harvard Mark I (IBM ASSC) machine, and its successor the Mark II are often excluded from the mainstream history of computers on two technicalities. The first is that Mark I and Mark II were electro-mechanical rather than electronic; the second one is that Aiken was never convinced that computer programs should be treated as data in what has come to be known as the von Neumann concept, or the stored program. It is not proposed to discuss here the origins and significance of the stored program. Nor I wish to deal with the related problem of whether the machines before the stored program were or were not “computers”. This subject is complicated by the confusion in actual names given to machines. For example, the ENIAC, which did not incorporate a stored program, was officially named a computer: Electronic Numeral Integrator and Computer. But the first stored-program machine to be put into regular operation was Maurice Wiles' EDSAC: Electronic Delay Storage Automatic Calculator. It seems to be rather senseless to deny many truly significant innovations (by H.H.Aiken and by Eckert and Mauchly), which played an important role in the history of computers, on the arbitrary ground that they did

not incorporate the stored-program concept. Additionally, in the case of Aiken, it is significant that there is a current computer technology that does not incorporate the stored programs and that is designated as (at least by TEXAS INSTRUMENTS®) as “Harvard architecture”, though, it should more properly be called “Aiken architecture”. In this technology, the program is fixed and not subject to any alteration save by intent - as in some computers used for telephone switching and in ROM. Aiken was a visionary, a man ahead of his times. Grace Hopper and others remember his prediction in the late 1940s, even before the vacuum tube had been wholly replaced by the transistor, that the time would come when a machine even more powerful than the giant machines of those days could be fitted into a space as small as a shoebox. Some weeks before his death Aiken had made another prediction. He pointed out that hardware considerations alone did not give a true picture of computer costs. As hardware has become cheaper, software has been apt to get more expensive. Then he gave us his final prediction: “The time will come”, he said, “When manufacturers will gave away hardware in order to sell software”. Time alone will tell whether or not this was his final look ahead into the future.

In the early 1960s, when computers were hulking mainframes that took up entire rooms, engineers were already toying with the then - extravagant notion of building a computer intended for the sole use of one person. by the early 1970s, researches at Xerox's Polo Alto Research Centre (Xerox PARC) had realized that the pace of improvement in the technology of semiconductors - the chips of silicon that are the building blocks of present-day electronics - meant that sooner or later the PC would be extravagant no longer. They foresaw that computing power would someday be so cheap that engineers would be able to afford to devote a great deal of it simply to making non-technical people more comfortable with this new information - handling tools. In their labs, they developed or refined much of what constitutes PCs today, from “mouse” pointing devices to software “windows”. Although the work at Xerox PARC was crucial, it was not the spark that took PCs out of the hands of experts and into the popular imagination. That happened inauspiciously in January 1975, when the magazine *Popular Electronics* put a new kit for hobbyists, called the Altair, on its cover. For the first time, anybody with $400 and a soldering iron could buy and assemble his own computer. The Altair inspired Steve Wosniak and Steve Jobs to build the first Apple computer, and a young college dropout named Bill Gates to write software for it. Meanwhile, the person who deserves the credit for inventing the Altair, an engineer named Ed Roberts, left the industry he had spawned to go to medical school. Now he is a doctor in small town in central Georgia. To this day, researchers at Xerox and elsewhere pooh-pooh the Altair as too primitive to have made use of the technology they felt was needed to bring PCs to the masses. In a sense, they are right. The Altair incorporated one of the first single-chip microprocessor - a semiconductor chip, which contained all the

basic circuits needed to do calculations - called the Intel 8080. Although the 8080 was advanced for its time, it was far too slow to support the mouse, windows, and elaborate software Xerox had developed. Indeed, it wasn't until 1984, when Apple Computer's Macintosh burst onto the scene, that PCs were powerful enough to fulfil the original vision of researchers. “The kind of computing that people are trying to do today is just what we made at PARC in the early 1970s,” says Alan Kay, a former Xerox researcher who jumped to Apple in the early 1980s.

Researchers today are proceeding in the same spirit that motivated Kay and his Xerox PARC colleagues in the 1970s: to make information more accessible to ordinary people. But a look into today's research labs reveals very little that resembles what we think of now as a PC. For one thing, researchers seem eager to abandon the keyboard and monitor that are the PC's trademarks. Instead they are trying to devise PCs with interpretive powers that are more humanlike - PCs that can hear you and see you, can tell when you're in a bad mood and know to ask questions when they don't understand something. It is impossible to predict the invention that, like the Altair, crystallizes new approaches in a way that captures people's imagination.

The Development of Computers in Ukraine and the Former USSR

The government and the authorities had paid serious attention to the development of the computer industry right after the Second World War. The leading bodies considered this task to be one of the principal for the national economy. Up to the beginning of the 1950s, there were only small productive capacities which specialized in the producing accounting and account-perforating (punching) machines. The electronic numerical computer engineering was only arising and the productive capacities for it were close to the naught. The first serious steps in the development of production base were made initially in the late 1950s when the work on creating the first industry samples of the electronic counting machines was finished and there were created M-20, “Ural-1”, “Minsk-1”, which together with their semi-conductor successors (M-220, “Ural-11-14”, “Minsk-22” and “Minsk-32”) created in the 1960s were the main ones in the USSR until the computers of the third generation were put into the serial production, that is until the early 1970s. In the 1960s, the science-research and assembling base was enlarged. As the result of this measures, all researches connected with creating and putting into the serial production of semi-conductor electronic computing machines were almost finished. That allowed stopping the production of the first generation machines beginning from the 1964. Next decades the whole branch of the computer engineering had been created. The important steps were undertaken to widen the productive capacities for the 3rd generation machines.

MESM was conceived by S.A.Lebedev to be a model of a Big Electronic Computing Machine (BESM). At first it was called the *Model* of the Big Electronic Computing Machine, but, later, in the process of its creation there appeared the evident expediency of transforming it in a small computer. For that reason there were added: the impute-output devices, magnetic drum storage, the register capacity was enhanced; and the word "*Model*" was changed for "*Malaya*" (Small). S.A.Lebedev was proposed to head the Institute of Energetic in Kiev. After a year; when the Institute of was divided into two departments: the electronic one and the department of heat-and-power engineering, Lebedev became the director of the first one. He also added his laboratory of analogue computation to the already existing ones of the electronic type. At once, he began to work on computer science instead of the usual, routine researches in the field of engineering means of stabilization and structures of automated devices. Lebedev was awarded the State Prize of the USSR. Since autumn 1948, Lebedev directed his laboratory towards creating the MESM. The most difficult part of the work was the practical creation of MESM. It might be only the many-sided experience of the researches that allowed the scientist to fulfil the task perfectly; whereas one inaccuracy was made: the hall at the ground-floor of a two-storied building was assigned for MESM and when, at last, the MESM was assembled and switched on, 6,000 of red-hot electronic lamps created the "tropics" in the hall, so they had to remove a part of the ceiling to decrease the temperature. In autumn 1951, the machine executed a complex program rather stabile. Finally, all the tests were over and on December 15, the MESM was put into operation. If to remember those short terms the MESM was projected, assembled, and debugged - in two years - and taking into consideration that only 12 people (including Lebedev) took part in the creating who were helped by 15 engineers we shall see that S.A.Lebedev and his team accomplished a feat (200 engineers and many workers besides 13 main leaders took part in the creation of the first American computer ENIAC). As life has shown, the foundations of the computer building laid by Lebedev are used in modern computers without any fundamental changes. Nowadays they are well known:

- such devices an arithmetic and memory input-output and control ones should be a part of a computer architecture;
- the program of computing is encoded and stored in the memory as numbers;
- the binary system should be used for encoding the numbers and commands;
- the computations should be made automatically basing on the program stored in the memory and operations on commands;
- besides arithmetic, logical operations are used: comparisons, conjunction, disjunction, and negation;
- the hierarchy memory method is used;

- the numerical methods are used for solving the tasks.

The great accumulated experience in creating computers, the profound comparison of our domestic achievements with the new examples of foreign computer technique prompted the scientists that it is possible to create the computing means of new generation meeting the world standards. Of that opinion were many outstanding Ukrainian scientists of that time - Lebedev, Dorodnitsin, Glushkov and others. They proceeded from quite a favourable situation in the country. The computerization of national economy was considered as one of the most essential tasks. The decision to create the united system of computers - the machines of new generation on integrals. The USA was the first to create the *families* of computers. In 1963-64, the IBM Company worked out the IBM-360 system. It comprised the models with different capacities for which a wide range of software was created. A decision concerning the third generation of computers (their structure and architecture) was to be made in the USSR in the late 60s. Instead of making the decision based on the scientific grounds, concerning the future of the United system of computers the Ministry of Electronic Industry issued the administrative order to copy the IBM-360 system. The leaders of the Ministry did not take into consideration the opinion of the leading scientists of the country. Despite the fact that there were enough grounds for thinking the 70s would bring new big progresses, those years were the step back due to the fault way dictated by the highest authorities from above.

E-mail

Email is cheaper and faster than a letter, less intrusive than a phone call, less hassle than a FAX. Using email, differences in location and time zone are less of an obstacle to communication. There is also evidence that email leads to a more egalitarian information structure. Because of these advantages, email use is exploding. By 1998, 30 per cent of adults in the United States and Canada had come on-line.

Electronic communication, because of its speed and broadcasting ability, is fundamentally different from paper-based communication. Because the turnaround time can be so fast, email is more conversational than traditional paper-based media. In a paper document, it is absolutely essential to make everything completely clear and unambiguous because your audience may not have a chance to ask for clarification. With email documents, your recipient can ask questions immediately. Email thus tends, like conversational speech, to be sloppier than communications on paper. This is not always bad. It makes little sense to slave over a message for hours, making sure that your spelling is faultless, your words eloquent, and your grammar beyond reproach, if the point of the message is to tell your co-worker that you are ready to go to lunch.

However, your correspondent also will not have normal status cues such as dress, diction, or dialect, so may make assumptions based on your name, address, and - above all - facility with language. You need to be aware of when you can be sloppy and when you have to be meticulous. Email also does not convey emotions nearly as well as face-to-face or even telephone conversations. It lacks vocal inflection, gestures, and a shared environment. Your correspondent may have difficulty telling if you are serious or kidding, happy or sad, frustrated or euphoric.

Another difference between email and older media is that what the sender sees when composing a message might not look like what the reader sees. Your vocal cords make sound waves that are perceived basically the same by both your ears as your audience's. The paper that you write your love note on is the same paper that the object of your affection sees. But with email, the software and hardware that you use for composing, sending, storing, downloading, and reading may be completely different from what your correspondent uses. Your message's visual qualities may be quite different by the time it gets to someone else's screen.

How E-mail Works

These days we all send email all the time - but have you ever wondered just how an email gets from your computer to its destination?

E-mail is one of the oldest internet technologies and it is the one that has truly changed the way we communicate. While we all use it, few stop to consider it, unless it's to complain about junk email. In fact, email is one of the easiest computing concepts to understand and, once you grasp the basics, you will soon recognize the wide-ranging possibilities of email and, crucially, realize why things can go wrong.

Email is easy to use, cheap and widely available. In the days when a single central computer was time-shared across a basic network, it was natural for programmers to invent programs that would allow users on different terminals to communicate with each other. At first this was hardly a world-changing facility because it allowed only a handful of people to communicate, but it established the idea of networked text messaging.

The origins of email as we know it today started with the first wide area network (WAN), ARPANET, a military network that was established in 1970. Extending the messaging facilities found on time-sharing machines into a WAN, which would eventually become the internet, was an obvious step. In 1971, an American computer engineer, Ray Tomlinson, implemented two small programs, SNDMSC and READMAIL. From here, things took off rapidly because it seemed that everyone wanted to email everyone else.

Surprisingly, the internet wasn't the first technology to provide email to the wider public. Such was the demand for email that the old-fashioned time-share computer architecture was brought back into play to provide email and other communications facilities such as bulletin boards. Users simply dialed up the service-providing computer and connected using a modem. While connected they could send and receive email.

You were restricted to sending email to other users of the same service and they had to dial up to collect and send email too. Although this was primitive, it attracted users. The main problem was the number of different systems available and the simple fact that they could not communicate with one another. In the USA, MCI Mail became the dominant email system; in the UK thousands of people had mailboxes on Telecom Gold. As public availability of the internet approached, these disparate systems tried without success to come together.

When the public was given access to the internet, email became a force that could change the way things were done. As the internet became a global phenomenon, for the first time you could send an email from almost any part of the world to almost any other part of the world. What's more, the cost of delivery bore no relation to the distance that the email had to go and the amount of information being exchanged. This is something we take for granted today but back then people were used to being charged by distance and quantity for communication of any sort.

Simple Mail Transport Protocol

Internet email was developed from existing email systems but it was also new, as it defined a standard that allowed email to be exchanged between any machines connected to the internet. The standard was simple - so simple, in fact, that they built the word 'simple' into its name. Simple Mail Transport Protocol (SMTP) is the basis of the worldwide email system. This is how it works. Some machines on the internet act, in effect, as post office letterboxes.

We take free email for granted today, but when it first appeared, people were used to being charged by distance and quantity for communication

An email file is transferred from the user to the nearest letterbox using the Simple Mail Transport Protocol (SMTP). This file contains some standard formatting that tells the receiving SMTP server where to send it, along with extra information detailing where it comes from. This information takes the form of 'headers' and you can see these in the email you receive, although there are usually far more than your email program shows you by default. The SMTP server looks at the 'To' header and, using the internet's standard address look-up system, finds the address of the SMTP server that controls the domain specified there.

So what is a domain? Well, in an email address the part before the @ specifies the person's mailbox and the part after the @ specifies the domain,

which you can think of as the address of the local sorting office. For example, if you send a message to:

mike@computershopper.co.uk

the email goes to computershopper.co.uk's SMTP server, which recognizes that the user 'mike' exists and moves the message to his mailbox for him to download later. Continuing the post office analogy, this mailbox is the equivalent of a Post Office (PO) box. You have to collect your messages - they are not delivered to your door. The SMTP server from which this email originates uses the domain name system (DNS) to locate computershopper.co.uk's server and transfers the email to it. That server then puts it in the correct PO box.

Many people use the SMTP server provided by their ISPs. If it is secured properly, only subscribers will be able to send mail via this server. If not, it can be used by spammers and other people who want to remain anonymous or difficult to trace. If you change ISPs, you might find that you can't send e-mail any more. Check that you are not trying to send mail through your old ISP's SMTP server.

SMTP and POP

SMTP is the protocol used to send mail out to the internet and is also used by one post office to transfer mail to another. Post Office Protocol (POP), on the other hand, is the protocol that allows you to retrieve mail. Your email client connects to your local POP post office machine using your user name and a password. Your mail is then collected from your PO Box. People often configure their email clients to send messages at the same time, using our old friend SMTP. All you need to make it all work is an email client that speaks POP3 (the current version of POP) and access to an SMTP server that accepts mail from your network. Your ISP's SMTP server will work fine.

The program that you use to deal with email, such as Outlook Express or Eudora, is a POP3 client. SMTP post office programs are less well known to users because they reside on servers maintained by internet service providers (ISPs). The most commonly used is Send Mail, which is part of most Linux distributions. Windows users tend to rely on Microsoft Exchange, which grew out of a proprietary email system, MSMail, that was developed before the internet became popular. Exchange supports both SMTP and POP3, but it also supports other legacy protocols and as a result is either more flexible than servers such as Send Mail or unnecessarily complicated, depending on your point of view.

As well as POP3, there is also Internet Message Access Protocol (IMAP), which can be thought of as an extension of POP3. This allows a client, in conjunction with a suitable server, to manage email without downloading it. You can, for example, read headers and delete email without downloading it first.

SMTP and POP3 are the foundation of email but things have developed beyond the basics. In the early days, text was enough for most users but people began to want to send other things: graphical files, sound files and even movies. This binary data needs to be encoded into ASCII text format before it can safely pass through email systems. Encoding methods include UUencode and Base 64, which both work in much the same way, coding the data in a binary file into a text representation. In those days people had to encode files manually using programs that were separate from the email client. The resulting ASCII was then pasted into a message and the recipient reversed the process.

The Multipurpose Internet Mail Extensions (MIME) facility was invented in 1992 to make sending file attachments easier. It establishes standard headers that are embedded in the email before the attachment and define the type of data that follows. The following attachment is encoded automatically by the mail client. Because this is a standard, any email client that can understand MIME will handle file attachments sent this way properly.

Perhaps the ultimate in fancy email is HTML mail. This works with MIME to allow an email to transfer a web page. Most email programs recognize this and show the attached web page as if it was an email message. This means that HTML mail can be used to send formatted emails that include pictures, sound files, and animations and so on.

Computer users are much better off since the SMTP, POP and MIME standards arrived. We can all send mail to each other, including many different types of attachments, and we have a reasonably good chance of it all working. However, this has come at the cost of security. There are serious defects with the openness of SMTP and POP3. You can use encryption to hide the contents of a message but there is no guarantee that it will be delivered to the right person. Even basic checks, such as asking for an auto-reply on receipt of email, are not part of the protocol. If you ask for confirmation of delivery, you will receive a reply only if the destination email client supports this feature and if the user has switched it on. Many other features that were taken for granted in proprietary email systems such as MSMail or Lotus cc:Mail are simply missing from internet mail.

There are also the twin problems of junk email, known as spam, and email viruses. The same mechanism that allows you to email a picture allows a virus writer to send you a virus. Some say the solution is to go back to text-only, non-MIME, non-HTML email, which would mean that there was nowhere to hide a virus. Spam is arguably a bigger problem; statistics are frequently published telling us about the many working hours that are lost as people are forced to delete unwanted and unsolicited emails. Sadly, email is so cheap and so insecure that spammers can send huge quantities of email while remaining anonymous or hiding behind someone else's identity. Perhaps this is the price we must pay for the ability to send an email to anyone, anywhere.

E-Business Up-to-date

Large corporations are using e-tools to deliver personalized customer service on a global scale.

E-business is not just about building online sales and supply networks. Many firms are embracing Customer-Relationship Management (CRM) technology in an effort to better respond to their customers' needs. In the long term, CRM can help a global corporation develop the kind of familiarity with its. Customers that is usually associated with small businesses, explains Herb Hunt, a former vice president for CRM at IBM. "One of the inherent advantages of a small company is that it's local," he says. "With CRM, it is possible to make a large corporation seem as intimate to the customer as if it were a small business down the street."

By aggregating and analyzing data from across different departments and customer-facing channels, companies can use CRM technology to generate on-the-spot responses to customer inquiries— and even produce individually tailored product and service offerings—all on an international scale.

Bayer, for instance, is developing comprehensive CRM solutions for its global health care and chemical businesses. Clemens Kaiser, Bayer's head of e-commerce, explains, "Our aim is to provide the same level of customer service everywhere, to ensure that all of our customer information is kept in one place, so that we will have globally uniform standards." However, Bayer's vision goes well beyond uniform standards of customer service. "Eventually, we would like to use predictive technology to pre-empt customers' demands and be in the marketplace earlier than our competitors with the right products," says Kaiser.

At Bayer, the issue is not so much new technology as it is how to use technology intelligently to create an enterprise wide customer database. "You have to use the technology cleverly, and we believe that we don't always need the latest gadget," says Kaiser. "We need a technology that works best for us in our diversified business and still allows us to use one common customer database and provide information to all of our employees in all our channels and at all customer touch points."

Few corporations can rival the customer base of British Airways, which served more than 48 million airline passengers last year. Eva Eisenschimmel, BA's general manager for customer knowledge and strategy is strongly committed to CRM—but she also believes that companies that rush to invest in the big-ticket CRM solutions may be falling into a trap. "The best CRM technology solutions are often to be found in connecting existing systems and databases with relatively low-cost options," she says. "CRM does not always have to involve spending vast sums of money."

While CRM originated as a high-tech marketing device, at many forward-looking companies it has evolved into a technique for better serving customer needs. Chris Shimizu, manager of corporate communications at NEC Corp., a leading global Internet solution provider, says NEC does not see CRM as a

means "for grasping the customer." Instead, Shimizu says, "We use CRM as a crucial tool in serving our customer bases. It's regarded as more of a demand-chain management tool as well."

At the Australia and New Zealand Banking Group (ANZ), CRM has been used to transform a onetime "customer-passbook mentality" into a new approach that "really looks at customers as customers," says Anthony Johnson, former head customer management in ANZ's personal finances division. "We realized that what we had acquired over time is a very rich base of customers that hadn't been properly 'data mined/ Back in 1994 the group started its journey on CRM by aggregating data in a group wide information warehouse. "As a result, ANZ has become far more adept at handling customer inquiries and financial needs. "Many organizations see CRM as relevant only to generating sales. What's interesting is that some banks have swung the pendulum the other way and are saying that the customer is absolutely the goal," he says.

While attracting and keeping customers has become the central focus of CRM at many firms, Bob Chatham, principal analyst, business applications research at Forrester Research, adds that there is one final step on the path to CRM: determining which of your satisfied customers are profitable, and which are not. "Ultimately, there are no unprofitable customers, only poorly managed companies," he concludes. “Firms must model customer behavior, turn analysis into action and revise constantly maximize each customer's profit."

REFERENCES

1. Japan Annual Reviews in Electronics, Computers & Telecommunications. Editor: T Kitagawa, OHM. – Tokyo, 1988.
2. Oryal Tanir. Modeling Complex Computer & Communication Systems. A Domain – Oriented Design Framework . McGraw. – Hill, 1997.
3. Sarah E. Hutchinson, Glen J. Goulthard. MICROSOFT Access 7.0 for Windows 95. Irwin, 1996.
4. Sarah E. Hutchinson. Microcomputer Application Software. Second edition. Irwin. – Boston, 1989.
5. Stephen L. Nelson. Windows NT4 for Busy People. Osborne / McGraw. – Hill, 1996.
6. Software Engineering: A Practitioner's Approach. Second Education. (Roger S. Pressman, Ph D). McGraw. – Hill Publishing Company, 1987.
7. Англо-русский политехнический словарь./ Сост. Ю. Синдеев. – Ростов н/Д: Фенікс, 2002.
8. Англо-український словник з інформатики та обчислювальної техніки./ Лінгв.ред. О.Р.Микитюк. – Львів: СП "БаК", 1995.
9. Англо-українсько-російський словник з інформатики, програмування, обчислювальної техніки. /А.Б. Бортіков. – К.: Вища шк., 1995.
10. Новый англо-русский словарь./В.К. Мюллер, В.Л. Дашевская, В.А. Каплан и др. – 3-е изд., стереот. – М.: Рус.яз., 1996.
11. Л.П. Зайцева, М.А. Бух. Микроэлектроника: настоящее и будущее.- М.: "Высшая школа", 1990.
12. В.І. Карабан. Переклад англійської наукової і технічної літератури. – Вінниця: Нова книга, 2001.

Навчальне видання

Н.В. Рибко, Н.А. Насонова

ТЕРМІНОЗНАВСТВО

ЧАСТИНА 2

Оригінал-макет підготовлено Н.В. Рибко

Редактор В.О. Дружиніна

Навчально-методичний відділ ВНТУ
Свідоцтво Держкомінформу України
серія ДК № 746 від 25.12.2001
21021, м. Вінниця, Хмельницьке шосе, 95, ВНТУ

Підписано до друку
Формат 29,7х42 ¼
Друк різографічний
Тираж прим.
Зам. №

Гарнітура Times New Roman
Папір офсетний
Ум. друк. арк.

Віддруковано в комп'ютерному інформаційно-видавничому центрі
Вінницького національного технічного університету
Свідоцтво Держкомінформу України
серія ДК № 746 від 25.12.2001
21021, м. Вінниця, Хмельницьке шосе, 95, ВНТУ