**Unit 15**

**PROGRAMMING**

***Vocabulary Bank Unit 15***

***Task 1. Memorize the following words and word-combinations:***

1. a form design grid
2. add-on (n)
3. assumption
4. bug
5. computation
6. constant
7. debugger
8. downside
9. event-driven
10. event-handling code
11. flow of execution
12. gracefully
13. inadequate
14. initial
15. iteration
16. language-specific symbols
17. loop
18. multiple
19. outline (n)
20. predefined
21. programming paradigm
22. pseudo code
23. regardless
24. remarks
25. repetition control structure
26. runtime error
27. screen-based object
28. self-contained
29. sequence of steps
30. step-by-step
31. structuring
32. to arrive a solution

|  |
| --- |
| 1. to deal with
 |

1. to define
2. to launch a program
3. user manuals
4. **VDE** (visual development environment)
5. verbal presentation
6. walkthrough
7. well-documented program
* **Introductory reading**

**Text 15 A. PROGRAM PLANNING**

Computers are controlled by sets of instructions called programs. Programs are written by a person called a programmer using special languages called programming languages.Programs can be written in a variety of computer languages. The language chosen will depend on a number of factors including what system the program will run on, what the function of the program is, and the knowledge of the programmer.

Programming is the process of preparing a set of coded instructions which enables the computer to solve specific problems or to perform specific functions. The essence of computer programming is the encoding of the program for the computer by means of algorithms. The thing is that any problem is expressed in mathematical terms; it contains formulae, equations and calculations. Any problem must be specially processed for the computer to understand it, that is – coded or programmed.

The phase in which the system's computer programs are written is called the development phase. The programs are lists of instructions that will be followed by the control unit of the central processing unit (CPU). The instructions of the program must be complete and in the appropriate sequence, or else the wrong answers will result. To guard against these errors in logic and to document the program's logical approach, logic plans should be developed.

There are two common techniques for planning the logic of a program. The first technique is flowcharting.

A flowchart is a plan in the form of a graphic or pictorial representation that uses predefined symbols to illustrate the program logic. It is, therefore, a "picture" of the logical steps to be performed by the computer. Each of the predefined symbol shapes stands for a general operation. The symbol shape communicates the nature of the general operation, and the specifics are written within the symbol. Flowcharts have arrowheads to indicate the direction of program flow and special symbols to indicate different functions in the program.

The second technique for planning program logic is called pseudocode. Pseudocode is an imitation of actual program instructions. It allows a program-like structure without the burden of programming rules to follow. Pseudocode is less time-consuming for the professional programmer than is flowcharting. It also emphasizes a top-down approach to program structure.

Pseudocode has three basic structures: sequence, decision, and looping logic. With these three structures, any required logic can be expressed.

The programming process begins with a problem statement that helps you clearly define the purpose of a computer program. In the context of programming, a problem statement defines certain elements that must be manipulated to achieve a result or goal. A good problem statement for a computer program has three characteristics:

1. It specifies any assumptions that define the scope of the problem.

2. It clearly specifies the known information.

3. It specifies when the problem has been solved.

In a problem statement an assumption is something you accept as true in order to proceed with program planning. The “known information” is the information that you supply to the computer to help it solve a problem. There are also variables (values that can change) and constants (factors that remain the same) in computer programs.

Formulating a problem statement provides a minimal amount of planning, which is sufficient for only the simplest programs. A typical commercial application requires far more extensive planning, which includes detailed program outlines, job assignments, and schedules. To some extent, program planning depends on the language and paradigm used to code a computer program. The phrase programming paradigm refers to a way of conceptualizing and structuring the tasks a computer performs. For example, whereas one programmer might focus on the steps required to complete a specific computation, another one might focus on the data that forms the basis for the computation. Quite a number of programming paradigms exist, and a programmer might use techniques from multiple paradigms while planning and coding a program.

There are different program planning tools, such as flowcharts, structured English, pseudocode, UML diagrams, and decision tables, which are used to provide sufficient planning.

Regardless of the tools used, when planning is complete, programmers can begin coding, testing, and documenting. The process of coding a computer program depends on programming language you use, the programming tools you select, and the programming paradigm that best fits the problem you are trying to solve. Programmers typically use a text editor, a program editor, or a VDE to code computer programs.

A text editor is any word processor that can be used for basic editing tasks, such as writing e-mail, creating documents, or coding computer programs. When using a text editor to code a computer program, you simply type in each instruction.

A program editor is a type of text editor specially designed for entering code for computer programs.

A VDE (visual development environment) provides programmers with tools to build substantial sections of a program by pointing and clicking rather than typing lines of code. A typical VDE is based on a form design grid that a programmer manipulates to design the user interface for a program. By using various tools provided by the VDE, a programmer can add objects, such as controls and graphics, to the form design grid. In the context of a VDE, a control is a screen-based object whose behavior can be defined by a programmer.

In visual development environment, each control comes with predefined set of events. Within the context of programming, an event is defined as an action, such as click, drag, or key press, associated with the form or control. A programmer can select the events that apply to each control. An event usually requires the computer to make some response. Programmers write event-handling code for the procedures that specify how the computer responds to each event.

A programmer’s choice of development tools depends on what is available for a particular programming language and the nature of the programming project. Text editors and program editors provide a fine tool set for programs with minimal user interfaces. A visual development environment is a powerful tool for programming software applications for GUI environments, such as Windows. Most GUI applications are “event-driven”, which means that when launched, the program’s interface appears on the screen and waits for the user to initiate an event.

A computer program must be tested to ensure that it works correctly. Testing often consists of running the program and entering test data to see whether the program produces correct results.

When a program doesn’t work correctly, it is usually the result of an error made by the programmer. A syntax error occurs when an instruction doesn’t follow the syntax rules, or grammar of the programming language. Syntax errors are easy to make, but they are usually also easy to detect and correct.

Another type of program bug is a runtime error, which, as its name indicates, shows up when you run a program. Some runtime errors result from instructions that the computer can’t execute.

Some runtime errors are classified as logic errors. A logic error is an error in the logic or design of a program. It can be caused by an inadequate definition of the problem or an incorrect formula for a calculation, and they are usually more difficult to identify than syntax errors.

Programmers can locate errors in a program by reading through lines of code, much like a proofreader. They can also use a tool called debugger to step through a program and monitor the status of variables, input, and output. A debugger is sometimes packaged with a programming language or can be obtained as an add-on.

Anyone who uses computers is familiar with program documentation in the form of user manuals and help files. Programmers also insert documentation called remarks or “comments” into the programming code. Remarks are identified by language-specific symbols.

A well-documented program contains initial remarks that explain its purpose and additional remarks in any sections of a program where the purpose of the code is not immediately clear.

***Task 2. Mark the following statements as True or False.***

1. The programming process begins with coding.
2. A typical commercial application requires a minimal amount of planning.
3. A programmer might use techniques from multiple paradigms while planning and coding.
4. Programmers typically use a program editor to code computer programs.
5. A visual development environment provides programmers with tools to build substantial sections of a program by pointing and clicking.
6. Text editors and program editors provide a fine tool for programming software interfaces.
7. Syntax errors result from instructions that the computer can’t execute.

***Task 3. Match up the words that are similar in meaning.***

computation medium

bug scheme

to execute error, mistake

environment calculation

outline carry out

to launch instrument

tool to start (up)

***Task 4. Fill in the blanks choosing from the variants given.***

1. Microsoft Visual Basic was one of the first programming languages to feature a visual development … .

a) medium b) environment c) tool

2. If program testing doesn’t produce the expected results, the program contains a (an) …, sometimes called a “...”.

a) mistake b) error c) problem d) bug

3. Program planning ... depends on the language and paradigm used to code a computer program.

a) instruments b) options c) tools

4. When the user … GUI application, the program interface appears on the screen and waits for the user to initiate an event by clicking a menu, dragging an object, or typing text.

a) starts up b) begins c) launches

5. A typical commercial program requires extensive planning, which includes detailed program … .

a) plans b) outlines c) schemes

6. Some runtime errors result from instructions that computer can’t ... .

a) execute b) make c) carry out

7. Programmers approach problems in different ways: while one programmer might focus on the steps to complete specific …, another programmer might focus on the data that forms the basis for the … .

a) tasks b) calculations c) computations

***Task 5. Make two-word combinations using the words in columns and then fill in the following sentences.***

A: programming B: planning

problem error

runtime statement

structured manual

event driven

program paradigm

user English

1. Anyone who uses computers is familiar with program documentation in the form of ... and help files.

2 The process of coding a computer program depends on programming language you use, the programming tools you select, and the … that best fits the problem you are trying to solve.

3. The programming process begins with a … that helps you clearly define the purpose of a computer program.

4. Most GUI applications are …, which means that when launched, the program’s interface appears on the screen and waits for the user to initiate an event.

5. An assumption is something you accept as true in order to proceed with … .

6. There are different program planning tools, such as flowcharts, … pseudocode, UML diagrams, and decision tables.

7. … shows up when you run a program.

***Task 6. Discuss the following questions.***

1. What is a problem statement?

2. What is an assumption?

3. Does the problem statement provide sufficient planning to begin coding?

4. How does a programmer code a computer program?

5. What is a text editor and a program editor?

6. What is a VDE?

7. How does a programmer know if a program works?

8. What can cause program errors?

9. How do programmers find errors?

10. Do computer programs contain any special documentation?

***Task 7. Read the text, write down the words you don’t know into your vocabulary and do the exercises below.***

**Text 15 B**

**PROCEDURAL PROGRAMMING**

The traditional approach to programming uses a procedural paradigm (sometimes called “imperative paradigm”) to conceptualize the solution to a problem as a sequence of steps. A program written in a procedural language typically consists of self-contained instructions in a sequence that indicates how a task is to be performed or a problem is to be solved.

A programming language that supports the procedural paradigm is called a procedural language. Procedural languages are well suited for problems that can be easily solved with a linear, or step–by-step, algorithm. Programs created with procedural languages have a starting point and an ending point. The flow of execution from the beginning to the end of the program is essentially linear – that is, the computer begins at the first instruction and carries out the prescribed series of instructions until it reaches the end of the program.

An algorithm is a set of steps for carrying out a task that can be written down and implemented. An algorithm for a computer program is a set of steps that explains how to begin with known information specified in a problem statement and how to manipulate that information to arrive a solution. In a later phase of the software development process, the algorithm is coded into instructions written in a programming language so that a computer can implement it.

To design an algorithm, you might begin by recording the steps you take to solve the problem manually. The computer also needs the initial information, so the part of your algorithm must specify how the computer gets it. Next, your algorithm should also specify how to manipulate this information and, finally, how the computer decides what to display as the solution.

You can express an algorithm in several different ways, including structured English, pseudocode, and flowcharts. These tools are not programming languages, and they cannot be processed by a computer. Their purpose is to give you a way to document your ideas for program design.

Structured English is a subset of the English language with a limited selection of sentence structures that reflects processing activities. Another way to express an algorithm is with pseudocode. Pseudocode is a notational system for algorithms that has been described as a mixture of English and your favorite programming language.

A third way to express an algorithm is to use a flowchart. A flowchart is a graphical representation of the way a computer should progress from one instruction to the next when it performs a task.

Before finalizing the algorithm for a computer program, you should perform a walkthrough to verify that your algorithm works. To perform a walkthrough for a simple program, you can use a calculator, paper, and pencil to step through a sample problem using realistic “test” data.

For more complex programs, a walkthrough might consist of a verbal presentation to a group of programmers who can help identify logical errors in the algorithm and suggest ways to make the algorithm more efficient.

The algorithm specifies the order in which program instructions are performed by the computer. Unless you do otherwise, sequential execution is the normal pattern of program execution. During sequential execution, the computer performs each instruction in the order it appears – the first instruction in the program is executed first, then the second instruction, and so on, to the last instruction in the program.

Some algorithms specify that a program must execute instructions in an order different from the sequence in which they are listed, skip some instructions under certain circumstances, or repeat instructions. Control structures are instructions that specify the sequence in which program is executed. Most programming languages have three types of control structures: sequence controls, selection controls, and repetition controls.

A sequence control structure changes the order in which instructions are carried out by directing the computer to execute an instruction elsewhere in the program. A sequence control structure directs the computer to the statements they contain, but when these statements have been executed, the computer neatly returns to the main program.

A selection control structure, also referred to as a “decision structure” or “branch”, tells a computer what to do, based on whether a condition is true or false. A simple example of a selection control structure is the IF…THEN…ELSE command.

A repetition control structure directs the computer to repeat one or more instructions until certain condition is met. The section of code that repeats is usually referred to as a loop or “iteration”. Some of the most frequently used repetition commands are FOR…NEXT, DO…WHILE, DO…UNTIL, and WHILE…WEND (which means “while ends”).

All the first programming languages were procedural. The first widely used standardized computer language, FORTRAN, with its procedural paradigm set the pattern for other popular procedural languages, such as COBOL, APL, ALGOL, PL/1, PASCAL, C, ADA, and BASIC.

The procedural approach is best suited for problems that can be solved by following a step-by-step algorithm. It has been widely used for transaction processing, which is characterized by the use of a single algorithm applied to many different sets of data. For example, in banking industry, the algorithm for calculating checking account balances is the same, regardless of the amounts deposited and withdrawn. Many problems in math and science also lend themselves to the procedural approach.

The procedural approach and procedural languages tend to produce programs that run quickly and use system resources efficiently. It is a classic approach understood by many programmers, software engineers, and system analysts. The procedural paradigm is quite flexible and powerful, which allows programmers to apply it to many types of problems.

The downside of the procedural paradigm is that it does not fit gracefully with certain types of problems – those that are unstructured or those with very complex algorithms. The procedural paradigm has also been criticized because it forces programmers to view problems as a series of steps, whereas some problems might better be visualized as interacting objects or as interrelated words, concepts, and ideas.

***Task 8. Indicate the paragraph where the following ideas are found in the text.***

1. A program written in a procedural language contains the prescribed series of instructions.
2. An algorithm shows the steps how to manipulate the information to arrive at a solution.
3. There are different tools to express an algorithm.
4. To make sure that your algorithm works, you should verify it.
5. Program instructions can be executed in order they are listed or some instructions can be skipped or repeated.
6. Many problems in banking industry lend themselves to the procedural approach.

***Task 9. Match up the words that are opposite in meaning.***

sequential parallel algorithm

downside problem

to focus written

solution advantage

to deposit to distract

linear algorithm random

verbal to withdraw

***Task 10. Fill in the blanks choosing from the variants given.***

1. During … execution,the computer performs each instruction in the order it appears – the first instruction in the program is executed first, then the second instruction, and so on, to the last instruction in the program.

a) random b) sequential c) direct d) reverse

2. The main ... of procedural paradigm is that it forces programmers to view problems as a series of steps, whereas some problems might better be visualized as interacting objects or as interrelated words, concepts, and ideas.

a) benefit b) advantage c) drawback d) downside

3. The fact that algorithms are usually written in a format that is not specific to a particular programming language allows you … on formulating a correct algorithm.

a) to concentrate b) to focus c) to distract

4. The traditional approach to programming uses a procedural paradigm to conceptualize the … a problem as a sequence of steps.

a) problem b) decision c) solution

5. The algorithm for calculating checking account balances is the same, regardless of the amounts … and.... .

a) invested, placed, deposited b) drawn out, withdrawn, taken away

6. Procedural languages are well suited for problems that can be easily solved with … algorithm.

a) chain b) linear c) parallel

7. For complex programs, a walkthrough might consist of a... presentation to a group of programmers who can help identify logical errors in the algorithm and suggest ways to make the algorithm more efficient.

a) written b) graphical c) verbal

***Task 11. Make three-word combinations using the words in columns and then fill in the gaps in the following sentences.***

A: selection B: account C: instruction

self step-by-step algorithm

software computer balances

standardized development language

checking control process

linear contained structure

1. The procedural approach is best suited for problems that can be solved by following a … .
2. A …, also referred to as a “decision structure” or “branch”, tells a computer what to do, based on whether a condition is true or false.
3. The first widely used …, FORTRAN, with its procedural paradigm set the pattern for other popular procedural languages.
4. In banking industry, the algorithm for calculating … is the same.
5. A program written in a procedural language typically consists of ... in a sequence that indicates how a task is to be performed or a problem is to be solved.
6. The algorithm is coded into instructions written in a programming language which a computer can implement in a later phase of the … .

***Task 12. Discuss the following questions.***

1. What is proceduralprogramming?

2. What is an algorithm?

3. How do you write an algorithm?

4. What is the best way to express an algorithm?

5. How do you know if your algorithm is correct?

6. In what order does a computer perform program instructions?

7. Can the computer make decisions while it executes a program?

8. What are the most popular procedural languages?

9. What kinds of problems are best suited to the procedural approach?

10. What are the advantages and disadvantages of the procedural paradigm?

***Task 14. Translate the following sentences into English.***

1. Якщо вам вдалося написати програму, в якій транслятором не виявлено помилок, зверніться до системного програмісту - він виправить помилки в трансляторі.

2. У природі програмування лежить те, що немає співвідношення між "розмірами" самої помилки і проблем, які вона спричиняє.

3. Якщо налагодження - процес видалення помилок, то програмування має бути процесом їх внесення.

4. Машинна програма виконує те, що ви наказали їй робити, а не те, що б ви хотіли, щоб вона робила.

5. Складність програми зростає до тих пір, поки не перевищить здатності програміста.

6. Якби архітектори будували будівлі так, як програмісти пишуть програми, то перший залетівший жук зруйнував би цивілізацію.

7. Ніколи не виявляйте в програмі помилки, якщо ви не знаєте, що з ними робити далі.

**WORD FORMATION**

**COMPOUND WORDS.**

There are three forms of compound words:

* the **closed form**, in which the words are melded together, such as firefly, secondhand, softball, childlike, crosstown, redhead, keyboard, makeup, notebook;
* the **hyphenated form**, such as daughter-in-law, master-at-arms, over-the-counter, six-pack, six-year-old, mass-produced;
* and the **open form**, such as post office, real estate, middle class, full moon, half sister, attorney general.

***Exercise 1. With a handful of exceptions, compounds created by the addition of a prefix are not hyphenated. Read and translate the following words.***

antisocial, binomial, biochemistry, coordinate, counterclockwise, extraordinary, infrastructure, interrelated, intramural, macroeconomics, metaphysical, microeconomics, midtown, minibike, multicultural, neoromantic, nonviolent, overanxious, postwar, preconference, pseudointellectual, reunify, semiconductor, socioeconomic, subpar, supertanker, transatlantic, unnatural, underdeveloped

**Exceptions include**

compounds in which the second element is capitalized or a number:

anti-Semitic, pre-1998, post-Freudian

compounds which need hyphens to avoid confusion

un-ionized (as distinguished from unionized), co-op

compounds in which a vowel would be repeated (especially to avoid confusion)

co-op, semi-independent, anti-intellectual (but reestablish, reedit)

compounds consisting of more than one word

non-English-speaking, pre-Civil War

compounds that would be difficult to read without a hyphen

pro-life, pro-choice, co-edited

***Exercise 2. Read and translate the words. Mind the spelling of these words.***

2-year education

one-week vacation

African American

Air Force

all-city tournament

attorney general

blood pressure

blue-green dress

bull's-eye

database

daughter-in-law

English-speaking person

ex-wife

first-rate accommodations

football

grandmother

grant-in-aid

great-aunt half sister

high-level officials

Italian-American

Italian-American club

jack-in-the-box

lifelike

light year

mayor-elect

salesperson

secretary-treasurer

stockbroker

T-square

threefold

up-to-the-minute

V-formation

vice president

well-made clothes

worldwide inflation

X-ray

With a series of nearly identical compounds, we sometimes delay the final term of the final term until the last instance, allowing the hyphen to act as a kind of place holder, as in

* The third- and fourth-grade teachers met with the parents.
* Both full- and part-time employees will get raises this year.
* We don't see many 3-, 4-, and 5-year-old children around here.

***Exercise 3. Study the most popular compound computer terms. Translate them into Ukrainian.***

**GUIDELINES ON THE CURRENT TREATMENT OF COMPOUND COMPUTER TERMS.**

* **In the following list, the two-word forms (shown first) are still more common, but the one-word forms are starting to take hold.**

file name OR: filename screen saver OR: screensaver

home page OR: homepage spell checker OR: spellchecker

menu bar OR: menubar voice mail OR: voicemail

* **In the following list, the one-word forms (shown first) are more common, but the spaced or hyphenated forms are still being used.**

barcode OR: bar code logoff (n.) OR: log-off

handheld OR: hand-held BUT: log off (v.)

hardwired OR: hard-wired logon (n.) OR: log-on

offline OR: off-line BUT: log on (v.)

offscreen OR: off-screen touchpad OR: touch pad

online OR: on-line touchscreen OR: touch screen

onscreen OR: on-screen wordwrap OR: word wrap

* **c. In the following list, the two-word forms (shown first) are more common, but the hyphenated forms (which follow the standard rules) are also being used.**

dot matrix printers OR: dot-matrix printers

local area networks OR: local-area networks

wide area networks OR: wide-area networks

* **In the following list, the hyphenated forms (shown first) are more common, but the solid or spaced forms (if given) are used in materials aimed at industry insiders.**

dot-com drop-down menu OR: dropdown menu

pop-up window pull-down menu OR: pulldown menu

read-only memory ink-jet printer OR: inkjet printer

write-only files random-access memory OR: random access memory

* **The following compound words are solid except in a few special cases.**

backup (n. & adj.) lookup (n.) trackball

BUT: back up (v.) BUT: look up (v.) trackpad

Desktop newsgroup uplink (n. & v.)

downlink (n. & v.) newsreader upload (n. & v.)

download (n. & v.) BUT: news server userid (derived from user ID)

keyword palmtop whois (derived from who is)

laptop toolbar workstation

* **Compound words beginning with Web are usually two words.**

Web site Web server BUT: Webmaster

Web page Web browser Webcasting

Web surfer Web directory Webzine

Web index Web clipping Weblog

Web cam Web terminal Webinar

*NOTE: The term Web site is still most commonly written as two words with a capital W. However, along with a few other Web compounds, it has started to appear as a solid word without an initial cap (website). In order to maintain a consistent style, it is better to retain the capital W until a majority of these terms (such as the World Wide Web and the Web) lose their initial cap as well.*

* **Compound words beginning with the prefix e are usually hyphenated.**

e-banking e-credit e-tail OR: e-tailing

e-book e-currency e-text

e-business e-dress (an e-mail address) e-wallet

e-cash e-lance BUT: eBay

e-commerce e-learning eDonkey

e-conomy e-money eHarmony

*The term e-mail can still be seen as E-mail (the original form of the word) and also as email (without the hyphen), but the hyphenated form is still the one most commonly used. In order to maintain a consistent style, it is better to retain the hyphen in e-mail until many of the other e words start to drop the hyphen as well*.

* **The prefix i (which refers to the Internet) appears both with and without a hyphen when it is attached to a base word.**

iPod i-Lighter

iMac i-Newswire

iTunes i-flex solutions

iPhone i-Safe

iTools BUT: I-80 (here I stands for Interstate)

* **The prefix m (which refers to the use of mobile phones) is usually followed by a hyphen when it is attached to a base word.**

m-business m-commerce

**GRAMMAR REVIEW**

***Exercise 4. Open the brackets and give the comparative or superlative degree of the following adjectives and adverbs.***

 1. That is (incredible) story I have ever heard. 2. It is not always (bright) students who do well in tests. 3. Terylene shirts are (hard) wearing, but cotton shirts are much (comfortable). 4. Which is (deep), Lake Michigan or Lake Superior? 5. She is far (self-confident) than she used to be. 6. (tall) man among the guests is a basketball player. 7.I like both of them, but I think Kate is (easy) to talk to. 8. Most people are (well off) than their parents used to be. 9. She has a lot to be thankful for; but (sad) thing of all is that she does not realize it. 10. I want to buy a car – (powerful) one you have. 11. You look a lot (sad) than you did last time I saw you. 12. There is nothing (irritating) than locking yourself out of your own house. 13. Both roads lead to the city centre, but the left-hand one is probably a bit (short) and (direct). 14. As I get (old), I notice the policemen seem to be getting (young). 15 The boys in our school are much (good-looking) and a lot (good) at football than the boys in other schools in the town.

***Exercise 5. Put the adverbs in the right places.***

1. You are right. (absolutely) 2. I got to bed at twelve. (always) 3. Do you go to parties? (ever) 4. You can be sure of anything. (never) But you can trust me. (certainly) 5. They meet every weekend. (usually) 6. My friends invite me to the theatre. (occasionally). 7. I have had such a shock! (never) 8. They met again. (never) 9. I remember meeting those people. (definitely). 10. Something is happening. (definitely) 11. Does he tell you the truth? (always) 12. He tries to do his best. (always) 13. He talks sensibly. (never) 14. He’s late. (always) He was late for his own wedding. (even) 15. Can you be sincere? (ever) 16. Expensive remedies are useful (always), if not to the sick, then to the chemist. 17. We fans give up hope. (never) 18. Are you all right? (really)

***Exercise 6. Solve these problems.***

1. 47 and 34 is \_\_\_ 2. 33 multiplied by 4 is \_\_\_ 3. 45 times 4 is \_\_\_ 4. 112 minus 45 \_\_\_ 5. 90 divided by 6 is \_\_\_ 6. Divide 66 by 11, multiply by 5, add 20, and subtract 18; you’ve got \_\_\_

***Exercise 7. Put each verb in brackets into an appropriate verb form.***

Jim (1)………… (not/look) forward to the exam. He (2) …………..(study) for the past two months, and still (3)………..(not/feel) sure that he (4)………….(know) even half of what he should know. He (5)……….. (question) his teachers repeatedly about the material that (6)…………(appear) on past exams, but he still (7)…………..(not/be) convinced that anything he (8)………..(learn) (9)………..(be) relevant to this year’s exam. He (10)………….(still/study) at seven o’clock in the morning on the day of the exam; he (11)…………..(revise) all night long, and (12)………….(feel) exhausted. In fact he (13)………… (be) so tired that he (14)……………(fall) asleep in the middle of writing the exam, and (15)………..(wake up) only just in time to finish it.

***Exercise 8. Correct the mistakes.***

1 The new regulations will been announced in September. 2 Dо they have their dogs walking every day? 3 She has the piano tune twice а year. 4 The letters are bе posted first thing tomorrow morning. 5 When Mr Cobert arrived at the office, he realized his computer has bееn stolen. 6 The photographs will already bееn developed. 7 It's worth having the car servicing. 8 She is expects to win the November election. 9 They are having their garage painted when I called. 10 Your teeth is bе cheked every six months.

***Exercise 9. Choose the right variant.***

1. I'll leave some sandwiches in the fridge in case you are hungry when you *(come)* back.

2. They *(could, had, must)* to put off their trip because their daughter suddenly had fallen ill.

3. Slow down. You *(go)* too fast."

4. Oxford is *(a, the,* -) most popular tourist attraction *(in, at)* Britain after London and Stratford-on-Avon.

5. I'm sure we *(meet)* many years ago, but I *(not, remember)* where it was.

6. Europe and America *(separate)* by the Atlantic Ocean.

7. He is trying to find (a, *the,* -) job, but there isn't *(much, many)* work available at present.

8. She said that she *(try)* to ring up her mother several times the previous day.

9. We have to pick the fruit very early in the morning; otherwise we *(can't, mustn't, may not)* get it to the market in time.

10. Nobody (use) this room for ages.

11. Tom Smith *(write)* a book and Brown and Co *(publish)* it last month.

12. Tomorrow the office *(clean)* in the evening after the staff leave.

13. We shook ... hands with ... host, (a, *the, -)*

14. Are French children obliged to go to *(a, the,* -) school *(on, at)* Sundays?

15. There will always be a conflict between ... old and ... young. Young people want ... changes but ... old people want ... things to stay ... same, *(a, the,* -)

16. He was standing there in the rain and *(not, have)* even the sense to put up his umbrella.

17. *(May, should, must)* I use your phone to ring for a taxi? Oh, there's no need *(for, to, at)* it, my son *(drive)* you home.

18. When we *(take)* our exams, we'll have a holiday.

19. People believed that he *(kill)* by terrorists.

20. He *(use)* to spend a lot of time in his library.

**WRITING**

***IV. Read the program below and the text, then complete the sentences which follow.***

**NOTES:** 1) *comment lines* – строка комментария; 2) *parentheses* – круглые скобки; 3) *braces* – фигурные скобки; 4) *declaration statement* – оператор объявлений; 5) *assignment statement* – оператор присваивания; 6) *variable name* – имя переменной; 7) *function statement* – функциональный оператор, оператор функции; 8) *semicolon* – точка с запятой; 9) *a must* – необходимость, потребность, требование; 10) *to terminate* – завершать, заканчивать; *terminator* – завершающая запись; 11) *blank line* – пустая строка; 12) *to span* – охватывать, изменять

|  |
| --- |
|  /\*CALCULATE AVERAGES \*/ main ( ) { float a, b, c, d, average;  printf (“Enter three numbers: “); scanf (“%f %f %f “, &a, &b, &c); d=a+b+c; average=d/3.0; printf (“The average is %f “, average); } |

***Comment Lines***

A C source program consists of statements and comment lines. Comment lines are enclosed by the characters /\* (at the start of the comment) and \*/ (at the end of the comment).

***The Function main { }***

Every C program must have a function called main which must appear only once in a program. The parentheses following the word main must be present, but there must be no parameters included. The main part of the program is enclosed within braces { }, and consists of declaration statements, assignment statements, and other C functions. In the above program there are six statements within the braces: a declaration statement (the first statement of the main program starting with the word float), two assignment statements (the fourth and fifth statements starting with the variable names d and average), and three function statements, two to print information on the screen and one to scan the keyboard for input.

As C is free-form language, the semicolon (;) at the end of each line is a must. It acts as a statement terminator, telling the compiler where an instruction ends. Free form means that statements can be identified and blank lines inserted in the source file to improve readability, and statements can span several lines. However, each statement must be terminated with a semicolon. If you forget to include the semicolon, the compiler will produce an error, indicating the next line as the source of the error. This can cause some confusion, as the statement objected to can be correct, yet as a syntax error is produced.

***Variables and the Declaration Statement***

A variable is a quantity that is referred to be name, such as a, b, c, d and average in the above program. A variable can take on many values during program execution, but you must make sure that they are given an initial value, as C does not do so automatically. However, before variables can be used in a program, they must be declared in a type declaration statement.

**EXERCISE. *Fill* *in the gaps to complete the sentences.***

1.The Function ... must appear only once in a program. 2. /\* CALCULATE AVERAGES\*/ is a ... line. 3. The statement float a, b, c, d, average; is a ... statement. 4. The program below contains ... function statements. 5. The assignment statements are on lines ... and ... . 6. The main part of the program is enclosed within ... . 7. Each line of any C program must end with a ..., which acts as a statement ... . 8. If you forget to include the correct punctuation, the ... will produce a ... error. 9. A quantity referred to by name is known as a ... . 10. A ... statement must be used to declare variables.