Unit 2. Programming

Lesson 3

Whole-Class Activity

Task 1. Pre-Assessment

You are going to read questions about programming. Use your background knowledge to answer them. You have 5 minutes to complete this task.

RATIONAL CONCERN

1. How did programming look like fifty years ago?
2. What was the main breakthrough in programming?
3. How often is programming used nowadays?

Practical Concern

1. What programming languages do you know?
2. Do you have any experience using programming?
3. Where do we use programming?

Analytical Concern

1. What classes of programming languages do you know?
2. What are the criteria of such classification?
3. What would have happened if programming remained on its first level of development?

1. Can programming be used in a creative way? Give examples.
2. In what way is programming used in such areas as music and art?
3. How can programming help create computer games?

Creative Concern

Task 2. Reading

Read the text about computer hardware and software and answer the quiz questions. You have 20 minutes for this activity.

Computer Programming

Writing software or computer programs means describing how to do something. In its simplest form, programming is breaking a task down into small **steps**. In this respect, writing a computer program can be like composing music, building a house or creating other things. It has been even argued that in its current state, programming is an *Art*, not engineering.

Today, most people don't need to know how a computer works. Most people can simply **turn on** a computer or a mobile phone and point at some graphical image on the display, click button, and the computer does something. Basically, computers **perform** operations on objects. A microprocessor, which is the heart of a computer, is really primitive but very fast. It manipulates groups of **binary numbers** – numeric values represented in only two symbols: 0 and 1. Microprocessor operates binary numbers, representing parts of objects, and moves them around, **adds** pairs together, **subtracts** one from another, **compares** a pair, etc. As an example, computer display consists of a bunch of objects called pixels. Every pixel has a position which consists of the row and column it is in. Its colour is specified as three numbers – called RGB (Red, Green, Blue) values.

Machine language, or machine code, is represented in 1s and 0s and is **executed** directly by a processor. The pioneers of computers wrote code in machine language, but no one does it these days.

One step above machine language is **assembler language**. In assembler, the operations that the microprocessor performs are given names. Addresses in memory can also be given meaningful names. This is a big step over binary, but still very tedious to do any large software program with. It still has its place for little pieces of software that need to **interact** directly with the microprocessor.

```
DBProvider = " Database provider" DB.connect
        SelectSQL1 = " Select id, name, quantity
        QuerySQL1 = " where id between decode(name,
        QuerySQL2 = " group by id, name"
SelectQuery = SelectSQL1 & QuerySQL1 & QuerySQL2
Execute Query; Commit Transaction; Select new data
Form Navigation
  If KeyAscii = 13 Then Execute Query
  If Not Chr(KeyAscii) Like "#" And KeyAscii <> 8 Then
```

Most software written today is written in **high-level languages**, some of them are quite old. For example, COBOL, FORTRAN, and Lisp were written in the 1950s. Instructions written in high-level languages are converted into machine code with the help of specific software called **compiler**. Thus, high-level languages abstract away the specifics of the microprocessor in your computer and may use natural language elements making the process of **developing** a program simpler and more understandable.

(The text is borrowed and modified from http://www.bfoit.org/Intro_to_Programming/Programming.html as of 12th February, 2013)

1. What is computer programming?
    A. Convincing the computer to never freeze.
    B. Speeding up your computer.
    C. Setting the alarm on a computer.
    D. Telling the computer what to do through a special set of instructions.
2. Basically, computers perform operations on ….
    A. Things.
    B. Objects.
    C. Icons.
    D. Stages.
3. Microprocessor can understand only commands written in…
    A. English language.
    B. 1s and 0s.
    C. Any numbers.
    D. Words and symbols.
4. In assembler languages, the names are given to…
    A. Objects.
    B. Microprocessor.
    C. Operations and addresses.
    D. Memory.
5. How is the software that translates code into something meaningful which the computer can understand called?
    A. Compiler.
    B. Converter.
    C. Transliterator.
    D. Translator.
6. Which class of languages allows for the use of words and commands?
    A. Machine code.
    B. High-level language.
    C. Assembler language.
    D. Source code.

Task 3. Vocabulary Practice

Match the words with their definitions. You have 5 minutes for this task.

1. Programming	a. a set of instructions executed directly by a computer's central processing unit
2. Binary numbers	b. creating a set of instructions that computers use to perform specific operations
3. Machine code	c. a computer program that transforms source code written in a programming language into another computer language
4. Assembler language	d. a program which translates from assembly language to machine language format
5. Compiler	e. representation of numeric values using two symbols: 0 and 1
6. High-level language	f. programming language with strong abstraction from the details of the computer
7. Assembler	g. a low-level programming language in which there is a very strong correspondence between the language and the machine code instructions

Task 4. Vocabulary Practice

Read the following sets of words / phrases and find out how they relate to each other. You have 5 minutes for this task.

E.g. "Program" is a synonym for "software". "Create" is an antonym for "destroy".

Programming, primitive, high-level, divide, convert, stage, turn on, part, execute, language, add, general, whole, complicated, low-level, step, specific, code, perform, change, break down, turn off, instruction, coding, command, subtract.

Task 5. Language in Use

When you deal with computer hardware and software you often need to explain how some items work. Study the table in which the ways of explaining the function of an item are given. After doing so, match the parts of a computer with their functions and explain them using the information you have studied. You have 10 minutes for this task.

Function of an Item

The function of an item can be described in the following ways:

1. Used for + V ing
   e.g. CDs are used for saving information.
2. Present Simple
   e.g. Operating system controls the hardware.
3. The function of ..... is to ....
   e.g. The function of input devices is to get information into computer.
4. be responsible for V ing
   e.g. The CPU is responsible for interpreting codes.
5. allows you …, enables you to …
   e.g. Software allows you perform different operations.

1.	Housing	a) interpret every code it receives from the other computer components, and make it usable to operating system
2.	Motherboard	b) produce a paper copy of the information on your screen
3.	CPU	c) hold all the parts that make the computer operate
4.	Hard drive	d) produce sound and music when connected to the computer
5.	Mouse	e) process information and perform computations
6.	Speakers	f) wire all of the computer parts together
7.	Printer	g) point to and select items on your screen
8.	Computer	h) store the information

Differentiated Activity

Task 6. Group activity

In groups, work out the topical content of the following text. After that you will have to report your findings to the class. You have 10 minutes for this task.

Software creation

In order to create software you have to do more than just learning a programming language. A quick overview of the process includes the following stages:

1. Writing a program

This means writing the steps needed to perform the task, using the programming language you know. Set of instructions which you type to create a program is usually called source code.

2. Compiling the program

When you write a program using programming language, it's not yet in a form that the computer can use. In order to use a program, you usually have to convert it into machine code first. This conversion process is called compiling, or compilation. A program called a compiler does the compiling.

3. Running the program

After compiling the program into a form that the computer can use, next step is to make the computer perform the steps that you specified. This is called running the program, or executing it.

4. Debugging the program

The term “debugging” came about because the earliest computers were huge building-sized machines, and real-life insects sometimes flew into the machinery, so first computer engineers had to physically "debug" the computers. Nowadays the term refers purely to fixing errors and problems in source code of the program.

5. Repeating the whole process until the program is finished

This stage includes improving the program until it satisfies the demands of software engineers and customers. This stage can be quite long and demanding. Programmers commonly release a new version of their program every day for a couple of weeks after the initial release.

Group 1. Your task is to track information in the text and give definitions to the following terms:

Compilation, source code, program executing, compiler, initial release, debugging.

Group 2. Your task is to create a scheme of the process with details and examples for each stage, if possible.

Group 3. Inside your group, give detailed explanation for every stage of software creation. Decide, what stage is the most important. Are there any other stages, which were not mentioned in the text?

Task 7. 🔊 Listening

You are going to watch a video about the development of object-oriented programming. Choose whatever part you feel confident to complete or do them all. You have 10 minutes for the task. Use the following link to watch the video: http://bigthink.com/videos/why-i-created-c

Part 1. Fill in the gaps:

In the really old days, people had to write their (1)………… directly to work on the (2) ………. You could do pretty good work with that, but it was very (3) ……….. Then they figured out that you could build languages fit for humans for specific areas, and they built FORTRAN for engineers and (4) …………. and COBALT for (5) ……….

And then a group of Norwegian programmers thought about creating a language that is fit for humans for all domains, not just linear (6) ……… and business. And they built language called SIMULA. And that's where they introduced the (7) ....…. as the thing you have in the program to represent a (8) ….…. in your application world. And they went a little bit further and represented (9) ………. between classes. This became known as (10) ………. or data abstraction.

Part 2. Decide whether the following statements are true (T) of false (F). Justify your answer.

1. In the old days people wrote code using hardware. □
2. FORTRAN and COBALT fit for specific areas. □
3. SIMULA was created in the mid -`16s. □
4. Class represents a concept in an application world. □
5. Matrix, personnel record and dial buffer might become a class. □
6. SIMULA failed to represent relationships between classes. □
7. Vehicle is a kind of car. □
8. Object-oriented programming is also called data abstraction. □

Part 3. Retell the text giving as many details as possible. You may use the text from Part 1 or statements from Part 2 as scaffolds.

Task 8. Pair work

Work in pairs. Using the help box from Task 5, ask each other and explain the functions of the following keys:

Shift, Caps Lock, Tab, Alt, Ctrl, End, Pg Dn, Print Screen

You have 10 minutes to complete this task. You may refer to the following site for information: http://www.otsego.k12.oh.us/bradley/basics.htm

Task 9. Role-play. Pair Work

Student 1. Imagine that you are running a website which sells unusual programs. Your work as a call operator and your task is to explain to the clients the functions of a program and propose other software. Look at Chart A for information.

Student 2. You are interested in creative and simple software to install on your mobile device. Your friend gave you a list of recommended funny programs. Ask the operator about their functions. Look at Note B for information.

You have time until the end of the class.

Chart A

Virtual Music Composer 4.0	Virtual Music Composer produces its own completely new musical themes.	$19,95
Face Off Max 3.4.9.2	Face Off Max is software that enables you to create funny photos by transplanting any face to any body and share the fun with your friends.	$10,00
SpoofKit 1.0	Spoofkit lets you send your family or friends, joke/spoof emails. You will be able to send your brother an email from the Queen or a letter from Arnold Swarchenegger.	$17,85
Horse Racing Predictor 1.5	With the help of advanced algorithms based on neural networks this revolutionary software will predict horse racing results with great accuracy	$19,95
Personality type test 1.0	This test determines your psychological state and personality type.	$13,25
Free Spy Message 1.0.0.6	The SpyMessage is easy-to-use, reliable and powerful tool for protecting important information that you don’t want others to see. With SpyMessage you can encrypt and hide your text message into image without any changes in its resolution or size.	$15,50

Note B

1. Personality type – best option))
2. Face off – useful in Facebook
3. Spoofkit made me laugh many times
4. Don`t ever try Horse Racing Predictor!!

Home Assignment

Do Tasks 1-3 from Workbook section.

WORKBOOK

Task 1. Tiered Task

Part 1. Insert the following words / phrases in the gaps. Translate the sentences into your native language.

perform	compare	machine language	interact
subtract	convert	high-level language	develop

1. While easily understood by computers, ………….. is almost impossible for humans to use because it consists entirely of numbers.
2. It is very important to ……….. user-friendly interface and make software easy to use.
3. If you ……….. 2 from 7, the answer is 5.
4. CPU can ………….. various operations very fast which makes it a useful tool in many human activities.
5. Programs written in ……………. are translated into assembly language by a compiler.
6. GUI allows users to ……….. with electronic devices using images rather than text commands.
7. When you ………. a file, the copy of it will appear on the desktop.
8. It is hard to ………….. modern programming languages with machine code.

Part 2. Replace the words in bold with their synonyms.

1. The process of software development includes such **stages** as program writing, compiling, running and debugging.
2. Unlike modern devices, the earliest computers were **huge** building-sized machines.
3. Several versions of the program were launched after the **initial** release.
4. Today, most people don't need to know how a computer **works**.
5. In assembler, the operations that the microprocessor **performs** are given names.
6. **Instructions** written in high-level languages are converted into machine code with the help of compiler.
7. Nowadays the term refers purely to fixing **bugs** and problems in source code of the program.
8. Programming is **breaking** a task **down** into small steps.

Part 3. Fill in the gaps with derivatives from the words in capitals.

A programming language is used to write computer programs and 1) ……...................... A program is written as a series of human 2) ……..................... computer commands that can be read by a 3) ………… and translated into machine code so that a computer can run it. From the moment you turn on your computer, it is running programs and carrying out 4)………….. Each and every 5)……………that your computer performs has instructions that someone had to write in a 6) ………….language. Nowadays such languages, as Java and C++ with C# are continuing to gain7)………….. There have been many attempts to automate this process, and have computers write computer programs but the 8) ……………is such that for now, humans still write the best computer programs.	APPLY UNDERSTAND COMPILE INSTRUCT OPERATE PROGRAM POPULAR COMPLEX

(The text is borrowed and modified from http://cplus.about.com/od/introductiontoprogramming/p/programming.htm as of 4th April, 2013)

Task 2. Tiered Task

Read the text and do at least one part of the task after reading.

The Tower of Babel – A Comparison of Programming Languages

These days, programming languages are becoming more and more general and all-purpose, but they still have their specializations, and each language has its disadvantages and advantages. Here is the comparison of the most popular programming languages.

C++ is well-suited for large projects because it has an object-oriented structure. People can collaborate on one program by breaking it down into parts and having a small group or even one individual work on each part. The object-oriented structure also allows code to be reused a lot, which can cut down development time. C++ is also a fairly efficient language – although many C programmers will disagree.

C is a popular language, especially in game programming, because it doesn't have the extra packaging of the object-oriented C++. Programmers use C because it makes programs faster and smaller than programs written in C++. You might wonder, however, whether it's worth giving up the reusability of C++ to get the small increase in performance with C, especially when C++ can, where necessary, be written in a C programming style.

Pascal is primarily a teaching language. Few industrial programs are written in Pascal. Pascal tends to use keywords instead of braces and symbols, so it is a bit easier for beginners to understand than languages like C++. Still, not everyone thinks

Pascal is just for the schools. Borland, the huge compiler software company, has been pushing Delphi, which is an object-oriented version of Pascal, as an industrial strength programming language.

Fortran is a number-crunching program, and it is still used by scientists because the language allows variables of any size up to the memory limit of the machine. Fortran is especially convenient for engineers, who have to mathematically model and compute values to high precision. Fortran, however, isn't nearly as flexible as C or C++. Programming in Fortran is rigid and has strict rules, which sometimes makes reading Fortran programs difficult.

Java is a multi-platform language that is especially useful in networking. Of course, the most famous usage of Java is on the web, with Java applets, but Java is also used to build cross-platform programs that stand alone. Since it resembles C++ in syntax and structure, learning Java is usually quite easy for most C++ programmers. Java offers the advantages provided by object-oriented programming, such as reusability; on the other hand, it can be difficult to write highly efficient code in Java, and Swing, its primary user interface, is notoriously slow. Nevertheless, Java has increased in speed in recent years, and version 1.5 offers some new features for making programming easier.

(The text is borrowed and modified from http://www.cprogramming.com/langs.html as of 14th April, 2013)

Part 1. Compare C++ with other programming languages and tick the language which is better in the following aspects. Justify your answer.

		C ++	C
1.	Efficacy		✓
2.	Performance		
3.	Reusability		
	---------------------------	C ++	Pascal
4.	Easiness		
5.	Prevalence		
	--------------------------	C ++	Fortran
6.	Flexibility		
	--------------------------	C ++	Java
7.	Speed		
8.	Easiness		

Part 2. Fill in the table with the information from the text.

#	*Programming language*	*Sphere of Use*	*Advantages*	*Disadvantages*
1	***C*** ++	*Large projects*	*Reusability* *Object-oriented structure*	*Less efficient than C*

2	***C***			
3	***Pascal***			
4	***Fortran***			
5	***Java***			

Part 3. Write your own short description of any programming language, which wasn’t mentioned in the text.

Task 3. Internet Search

Visit the following link http://www.youtube.com/watch?v=DBXZWB_dNsw, or find any other source of information about prodigy programmers and be ready to retell their story. Mention the following information:

1. Name, country;
2. Age, education;
3. Field of interests;
4. Personality;
5. Story of programming success.