COM204 Module 1 AVP Transcript

Title Slide

Slide Content:

COM204 Module 1

Slide 1

Title: Understanding Computer Systems

Slide content:

- Computer system is a combination of all the components required to process and store data using a computer.
- Hardware is equipment associated with a computer.
- Software consists of computer instructions that tell the hardware what to do.
  - Instructions are developed and written by programmers

Narrator:
This module starts with a brief review the major components of a computer system. It differentiates between the hardware and software components of a computer system and identifies the roles they play. Then an overview of the program development process is examined.
A computer system is a combination of all the hardware and software components required to process and store data using a computer.

Hardware consists of the physical components of a computer – the tangible parts that you can see and touch. For example, keyboards, mice, speakers, and printers are all hardware.

When you think of a computer you often think of its physical components first. But for a computer to be useful, however, it needs more than the devices that form the hardware; a computer needs to be given instructions that control how and when data items are input, how they are processed, and the form in which they are output or stored.

Software consists of computer instructions that tell hardware what to do. These instructions are developed by programmers. In this course you will learn the techniques used by programmers to create such instructions.

One of the most important steps in writing software is planning the logical sequence of instructions to accomplish a task. This module explains the process of developing program logic that enables specifying a logical sequence of instructions in a program. For a program to work properly, you must develop the correct logic.

Slide 2

Title: Understanding Computer Systems
Programming is writing software instructions.
Computer hardware and software accomplish three major operations:
- Input
- Processing
- Output

The task of writing the logical sequence of instructions to perform a task is called programming. Computer hardware and software work in conjunction to accomplish three major tasks: input, processing, and output:
1. Input involves accepting data from a variety of possible input sources such as a keyboard, mouse, a data file, a computer network, a sensor input etc.
2. Processing involves using a set of instructions to perform a set of processing tasks on the data
3. Output represents presenting the results from the processing of data

Programming language is used to write computer instructions. Examples are Visual Basic, C++, C#, or Java.

Syntax is the set of rules governing its word usage and punctuation.

Computers instructions are written using computer programming languages such as Visual Basic, C++, C#, or Java. Why do we need programming languages to instruct computers? Why not use traditional languages such as English to issue such instructions? Computer instructions need to be very precise, with no room for ambiguity, and programming languages are designed to enable this.

Just as traditional languages such as English, every programming language has rules governing its word usage and punctuation. These rules are called the language’s syntax. All programs written in a programming language must conform to the language’s syntax for the program to work.

A compiler or an interpreter translates program language into machine code (binary code) and checks for syntax errors.
When the syntax and logic of a program are correct, you can run (or execute) the program to produce the desired results.
The logic for a computer program is the sequence of program instructions that represents a working program.

**Narrator:**
Once programs are written in a high-level programming language, they must be translated into binary code (called machine code) that the CPU can process. This translation is accomplished using a type of program translator called a compiler or an interpreter. The machine code produced by these translators can then be directly executed by the computer’s CPU. Such compiled machine code is often called an “executable” file.

When the syntax and logic of a program are correct, you can run, or execute the program to produce the desired results.

For a program to work properly, you must develop correct logic. The logic for a computer program is the sequence of program instructions that represents a working program.

Writing a computer program to solve a problem involves understanding a problem, then developing a solution, and finally writing the program using a programming language.

**Slide 5**

**Title:** Understanding the Program Development Cycle

**Slide Content:**
Program development cycle:
   1. Understand the problem
   2. Plan the logic
   3. Code the program
   4. Use software (a compiler or interpreter) to translate the program into machine language
   5. Test the program
   6. Put the program into production
   7. Maintain the program

**Narrator:**
Professional programmers use a well-defined methodology when developing complex software applications. This methodology follows a series of steps called a Program Development Cycle, that starts with a clear understanding of the problem specification, followed by planning the program logic, writing the program in a programming language, translating the program into machine language, testing the program, deploying the program, and finally maintaining the program.

**Slide 6**

**Title:** Planning the Logic

**Slide Content:**
   - Heart of the programming process
The most common planning tools are flowcharts and pseudocode.

Desk-checking is the process of walking through a program’s logic on paper before you actually write the program.

Narrator:
The planning phase in the program development cycle is the most important step in developing a program. This sets the stage for the rest of the program development process. During this phase of the process, the programmer plans the steps if the program, deciding what steps to include and how to order them.

When programmers plan the logic for a solution to a programming problem, they often use flowcharts or pseudo code to plan and document the program logic.

During the planning stage, the main focus is on figuring out what sequence of events will lead from the available input to the desired output. Planning the logic includes thinking carefully about all possible data values a program might encounter and how the program handles each scenario. The process of walking through a program’s logic on paper before actually writing the program is called desk-checking.

Slide 7

Title: Using Software to Translate a Program into Machine Language

Slide Content:
- Translator program is a compiler or interpreter that changes the programmer’s English-like high-level programming language into the low-level machine language
- Syntax error is the misuse of a language’s grammar rules
  - A programmer corrects listed syntax errors, and might need to recompile the code several times

Narrator:
Programs written in a high-level programming language such as C++ or Java must first be translated to the machine language that a computer’s hardware understands. The two types of translators commonly used to accomplish this are compilers and interpreters.

A compiler is a separate program that converts the entire source program into machine language before executing it. While it is compiling the source program, the compiler checks the source program for errors. The compiler then produces a program listing that contains the source code and a list of any errors. The listing helps the programmer make necessary changes to the source code and correct errors in the program.

A compiler translates an entire program before executing it. An interpreter, by contrast, translates and executes one statement at a time. An interpreter reads a program statement, converts it into one or more machine language instructions, and then executes those machine language instructions. It does all this before moving to the next statement in the program. Each time the source program runs, the interpreter translates and executes it, statement by statement.

Slide 8
Title: Program Planning Tools

Slide Content:
- Flowchart is a pictorial representation of the logical steps it takes to solve a problem
- Pseudocode is English-like representation of the logical steps it takes to solve a problem

Narrator:
Two of the commonly used tools to aid planning the logic for programs are flowcharts and pseudo code. Either of these two can be used interchangeably and the tool adopted is a matter of preference and convenience. Flowchart is a diagrammatic tool that uses a combination of geometric shapes and text to illustrate program logic. Pseudo code is a textual representation that uses English-like words and phrases to denote the logical steps used in a program.

Slide 9

Title: Flowchart vs. Pseudocode

Slide Content:
Examples of a flowchart and a pseudocode

Narrator:
This slide compares the use of a flowchart and a pseudo code to document the program logic for a simple illustrative program that accepts a number entered by a user, doubles its value, and then prints it. You can see from the figures on the slide that the flowchart and pseudo code are the same – only the presentation format differs.