

# Introduction to Computers and Programming

## Topics

- Introduction
- Hardware and Software
- How Computers Store Data
- How a Program Works
- Using Python

2

## Introduction

- Computers can be programmed
  - Designed to do any job that a program tells them to
- Program: set of instructions that a computer follows to perform a task
  - Commonly referred to as *Software*
- Programmer: person who can design, create, and test computer programs
  - Also known as software developer

3

## Hardware and Software

- Hardware: The physical devices that make up a computer
  - Computer is a system composed of several components that all work together
- Typical major components:
  - Central processing unit
  - Main memory
  - Secondary storage devices
  - Input and output devices

4

## The CPU

- Central processing unit (CPU): the part of the computer that actually runs programs
  - Most important component
  - Without it, cannot run software
  - Used to be a huge device
- Microprocessors: CPUs located on small chips

5

## Main Memory

- Main memory: where computer stores a program while program is running, and data used by the program
- Known as *Random Access Memory* or *RAM*
  - CPU is able to quickly access data in RAM
  - Volatile memory used for temporary storage while program is running
  - Contents are erased when computer is off

6

## Secondary Storage Devices

- Secondary storage: can hold data for long periods of time
  - Programs normally stored here and loaded to main memory when needed
- Types of secondary memory
  - Disk drive: magnetically encodes data onto a spinning circular disk
  - Solid state drive: faster than disk drive, no moving parts, stores data in solid state memory
  - Flash memory: portable, no physical disk

7

## Input Devices

- Input: data the computer collects from people and other devices
- Input device: component that collects the data
  - Examples: keyboard, mouse, touchscreen, scanner, camera
  - Disk drives can be considered input devices because they load programs into the main memory

8

## Output Devices

- Output: data produced by the computer for other people or devices
  - Can be text, image, audio, or bit stream
- Output device: formats and presents output
  - Examples: video display, printer
  - Disk drives and USB drives can be considered output devices because data is sent to them to be saved

9

## Software (1 of 2)

- Everything the computer does is controlled by software
  - General categories:
    - Application software
    - System software
- Application software: programs that make computer useful for every day tasks
  - Examples: word processing, email, games, and Web browsers

10

## Software (2 of 2)

- System software: programs that control and manage basic operations of a computer
  - Operating system: controls operations of hardware components
  - Utility Program: performs specific task to enhance computer operation or safeguard data
  - Software development tools: used to create, modify, and test software programs

11

## How Computers Store Data

- All data in a computer is stored in sequences of 0s and 1s
- Byte: just enough memory to store letter or small number
  - Divided into eight bits
  - Bit: electrical component that can hold positive or negative charge, like on/off switch
  - The on/off pattern of bits in a byte represents data stored in the byte

12

## Storing Numbers

- Bit represents two values, 0 and 1
- Computers use binary numbering system
  - Position of digit  $j$  is assigned the value  $2^{j-1}$
  - To determine value of binary number sum position values of the 1s
- Byte size limits are 0 and 255
  - 0 = all bits off; 255 = all bits on
  - To store larger number, use several bytes

13

## Storing Characters

- Data stored in computer must be stored as binary number
- Characters are converted to numeric code, numeric code stored in memory
  - Most important coding scheme is ASCII
    - ASCII is limited: defines codes for only 128 characters
  - Unicode coding scheme becoming standard
    - Compatible with ASCII
    - Can represent characters for other languages

14

## Advanced Number Storage

- To store negative numbers and real numbers, computers use binary numbering and encoding schemes
  - Negative numbers encoded using two's complement
  - Real numbers encoded using floating-point notation

15

## Other Types of Data

- Digital: describes any device that stores data as binary numbers
- Digital images are composed of pixels
  - To store images, each pixel is converted to a binary number representing the pixel's color
- Digital music is composed of sections called samples
  - To store music, each sample is converted to a binary number

16

## How a Program Works (1 of 3)

- CPU designed to perform simple operations on pieces of data
  - Examples: reading data, adding, subtracting, multiplying, and dividing numbers
  - Understands instructions written in machine language and included in its instruction set
    - Each brand of CPU has its own instruction set
- To carry out meaningful calculation, CPU must perform many operations

17

## How a Program Works (2 of 3)

- Program must be copied from secondary memory to RAM each time CPU executes it
- CPU executes program in cycle:
  - Fetch: read the next instruction from memory into CPU
  - Decode: CPU decodes fetched instruction to determine which operation to perform
  - Execute: perform the operation

18

## How a Program Works (3 of 3)

19

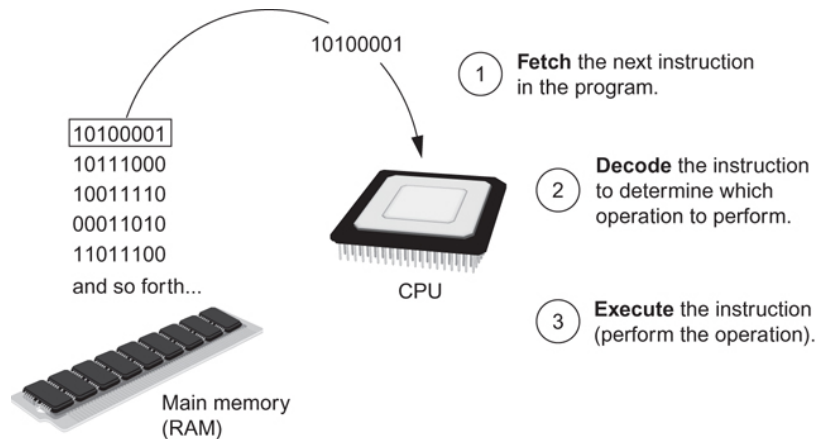


Figure 1-16 The fetch-decode-execute cycle

19

## From Machine Language to Assembly Language

- Impractical for people to write in machine language
- Assembly language: uses short words (mnemonics) for instructions instead of binary numbers
  - Easier for programmers to work with
- Assembler: translates assembly language to machine language for execution by CPU

20

## High-Level Languages

- Low-level language: close in nature to machine language
  - Example: assembly language
- High-Level language: allows simple creation of powerful and complex programs
  - No need to know how CPU works or write large number of instructions
  - More intuitive to understand

21

## Keywords, Operators, and Syntax: an Overview

- Keywords: predefined words used to write program in high-level language
  - Each keyword has specific meaning
- Operators: perform operations on data
  - Example: math operators to perform arithmetic
- Syntax: set of rules to be followed when writing program
- Statement: individual instruction used in high-level language

22

## Compilers and Interpreters (1 of 3)

- Programs written in high-level languages must be translated into machine language to be executed
- Compiler: translates high-level language program into separate machine language program
  - Machine language program can be executed at any time

23

## Compilers and Interpreters (2 of 3)

- Interpreter: translates and executes instructions in high-level language program
  - Used by Python language
  - Interprets one instruction at a time
  - No separate machine language program
- Source code: statements written by programmer
  - Syntax error: prevents code from being translated

24

## Compilers and Interpreters (3 of 3)

25

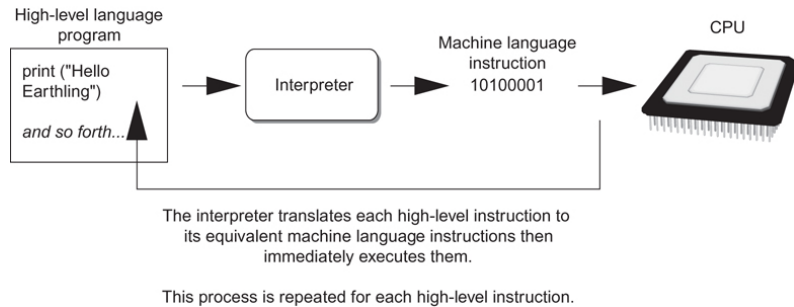


Figure 1-19 Executing a high-level program with an interpreter

25

## Using Python

- Python must be installed and configured prior to use
  - One of the items installed is the Python interpreter
- Python interpreter can be used in two modes:
  - Interactive mode: enter statements on keyboard
  - Script mode: save statements in Python script

26

## Interactive Mode

- When you start Python in interactive mode, you will see a prompt
  - Indicates the interpreter is waiting for a Python statement to be typed
  - Prompt reappears after previous statement is executed
  - Error message displayed if you incorrectly type a statement
- Good way to learn new parts of Python

27

## Writing Python Programs and Running Them in Script Mode

- Statements entered in interactive mode are not saved as a program
- To have a program use script mode
  - Save a set of Python statements in a file
  - The filename should have the `.py` extension
  - To run the file, or script, type

```
python filename
```

at the operating system command line

28

## The IDLE Programming Environment

- IDLE (Integrated Development Program): single program that provides tools to write, execute and test a program
  - Automatically installed when Python language is installed
  - Runs in interactive mode
  - Has built-in text editor with features designed to help write Python programs

29

## Summary

- This chapter covered:
  - Main hardware components of the computer
  - Types of software
  - How data is stored in a computer
  - Basic CPU operations and machine language
  - Fetch-decode-execute cycle
  - Complex languages and their translation to machine code
  - Installing Python and the Python interpreter modes

30