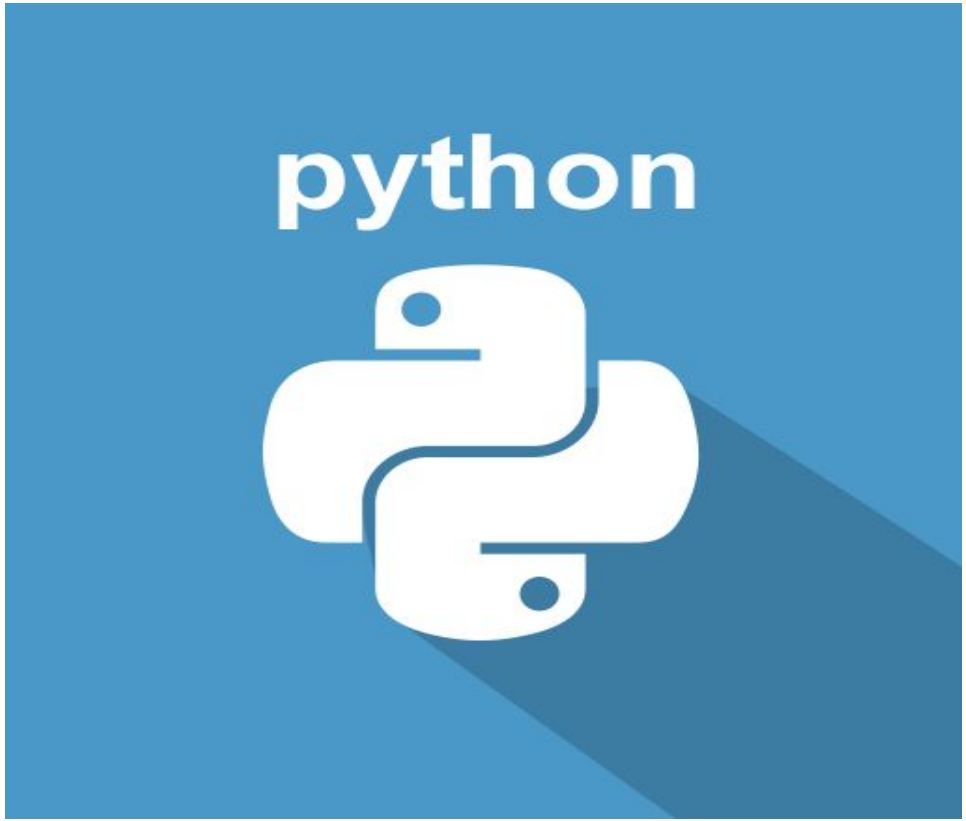
Lecture 1

Introduction to Computers and Python Programming



Who am I?

- Data Scientist
- Machine Learning Practitioner
- Systems Engineer
- Software Engineer
- Entrepreneur

Why Python?

FOSSBYTES

Netflix Reveals Python Is The Programming Language Behind The Films You Watch

Topics

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

Introduction

- **Computers can be programmed**
- **Program: set of instructions that a computer follows to perform a task**
  - Often referred to as *Software*
- **Programmer: person who can design, create, and test computer programs**

Hardware and Software

- **Hardware: The physical devices that make up a computer**
  - Computer is a system composed of several components that all work together

Hardware and Software

- **Typical major components:**
  - Central processing unit
  - Main memory
  - Secondary storage devices
  - Input and output devices

The CPU

- **Central processing unit (CPU): the part of the computer that actually runs programs**
  - Software is broken down into individual instructions
  - CPU generally runs one instruction at a time

The CPU

- **Microprocessors: Processing Units located on small chips**
  - Often help with auxiliary tasks (displaying graphics e.g.)

Main Memory

- **Main memory: where computer stores program instructions for quick retrieval**
  - Also referred to as RAM
  - CPU is able to quickly access data in RAM
  - Volatile memory used for temporary storage while program is running

Secondary Storage Devices

- **Secondary storage: can hold data for long periods of time**
  - Programs normally "permanently" stored here and loaded to main memory/RAM when needed

Secondary Storage Devices

- **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
  - Optical devices: data encoded optically

Input Devices

- **Input device: component that collects input data from people and other devices**
  - Examples: keyboard, mouse, scanner, camera

Output Devices

- **Output device: formats and presents output to user**
  - Examples: video display, printer, speaker

Software

- **Almost everything the computer does is controlled by software**
- **Software can be broken up into two general categories**
  - Application software
  - System software

Software

- **Application software: programs that make computer useful for every day tasks**
  - Examples: word processing, email, games, and Web browsers

Software

- **System software: programs that control and manage basic operations of a computer**
  - Operating system: provides controls for operations of hardware components
  - Utility Program: enhance computer operations and/or safeguard data

How Computers Store Data

- All data in a computer is stored in sequences of 0s and 1s
- Each 1 or 0 is known as a ***bit***
- 8 bits (enough to represent tokens like words) is known as a ***byte***

Storing Numbers

- **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

Storing Numbers

Bytes represent numbers in the range [0 … 255]

- 0 = all bits off; 255 = all bits on
- To store larger numbers you can use several bytes

Storing Characters

- Data stored in computer must be stored as binary number
- Characters are converted to numeric codes prior to being stored in memory

Storing Characters

- Historically the most prominent coding scheme was ASCII
  - ASCII is limited: defines codes for only 128 characters
- Unicode coding scheme has become standard
  - Can represent characters for many languages

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

Other Types of Data

- **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**
  - Audio samples are converted to binary sequences

How a Program Works

- **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

How a Program Works

- To carry out meaningful calculation, CPU must perform many operations

How a Program Works

1. Program must be copied from secondary memory to RAM each time CPU executes it

How a Program Works

2. **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

How a Program Works

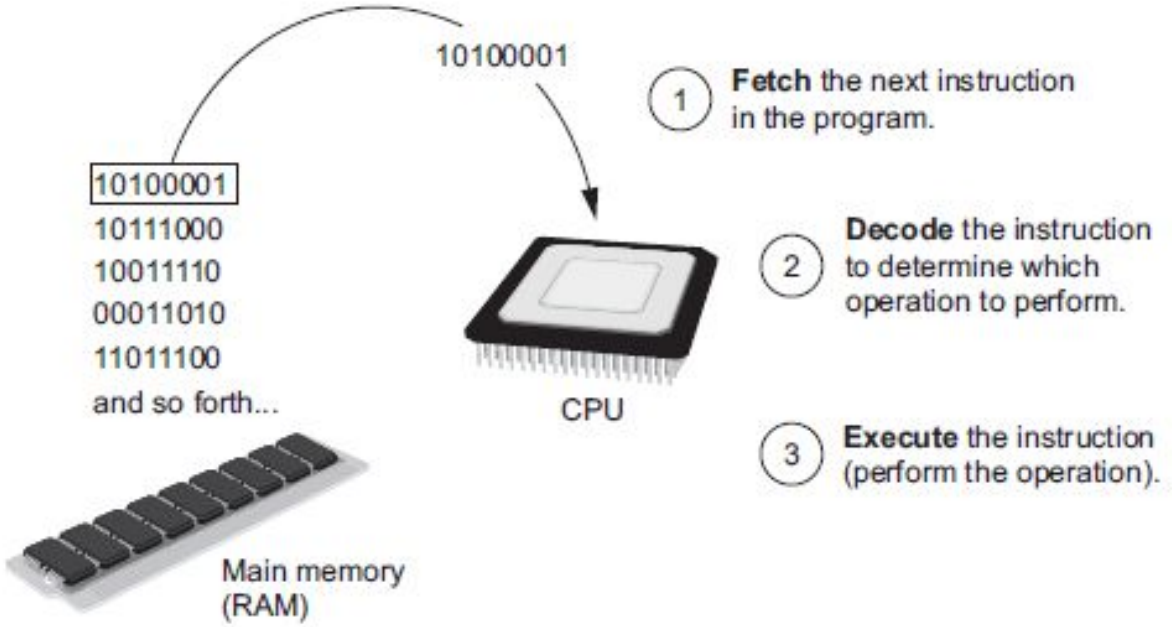


Figure 1: The fetch-decode-execute cycle

High-Level Languages

- **Low-level language: close in nature to machine 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

Key Words, Operators, and Syntax: an Overview

- **Key words: predefined words used to write program in high-level language**
  - Each key word has specific meaning
- **Operators: perform operations on data**
  - Example: math operators to perform arithmetic

Key Words, Operators, and Syntax: an Overview

- Syntax: set of rules to be followed when writing program
- Statement: individual instruction used in high-level language

Compilers and Interpreters

- Programs written in high-level languages must be translated into machine language to be executed

Compilers and Interpreters

- **Compiler: translates high-level language program into separate machine language program**
  - Machine language program can be executed at any time

Compilers and Interpreters

- **Interpreter: translates and executes instructions in high-level language program**
  - Interprets one instruction at a time
  - No separate machine language program

Compilers and Interpreters

- **Source code: statements written by programmer**
  - Syntax error: prevents code from being translated

Compilers and Interpreters

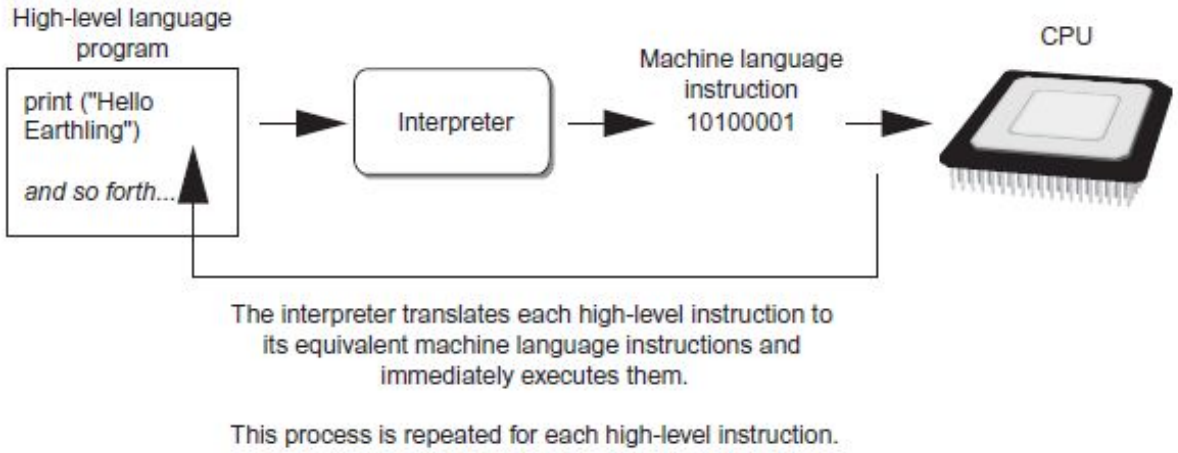


Figure 2: Executing a high-level program with an interpreter

Using Python

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

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

Python 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
  - To run the file, or script, type

    python *filename*

    at the operating system command line

Summary

- **We covered:**
  - Main hardware components of the computer
  - 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