Introduction to Computer Programming

Lecture 1
How Computers and Programs Work (briefly)
Lots of Jargon

• Programs are also called *Software, Code, Apps, Scripts* ...

• Programmers are also called *Coders, Software Engineers, Developers, ...*

• Programming is also called *Coding, Development, ...*

• Hardware: the physical components of the computer

If I use a term you don’t understand --- please ask what it means!
Figure 1-1  A word processing program and an image editing program

Software Components

1.35 What is the purpose of ASCII?
1.36 What encoding scheme is extensive to represent all the characters of all the languages in the world?
1.37 What do the terms “digital data” and “digital device” mean?

1.4 How a Program Works

Concept: A computer’s CPU can only understand instructions that are written in machine language. Because people find it very difficult to write entire programs in machine language, other programming languages have been invented.

Earlier, we stated that the CPU is the most important component in a computer because it is the part of the computer that runs programs. Sometimes the CPU is called the “computer’s brain,” and is described as being “smart.” Although these are common metaphors, you should understand that the CPU is not a brain, and it is not smart. The CPU is an electronic device that is designed to do specific things. In particular, the CPU is designed to perform operations such as the following:

- Reading a piece of data from main memory
- Adding two numbers
- Subtracting one number from another number
- Multiplying two numbers
- Dividing one number by another number
- Moving a piece of data from one memory location to another
- Determining whether one value is equal to another value
- And so forth...

As you can see from this list, the CPU performs simple operations on pieces of data. The CPU does nothing on its own, however. It has to be told what to do, and that’s the job of the operating system.
What is a Program?

• A program is a list of instructions that a computer can understand.
• To bring a program to life, Run it (Execute it)....

Input
1223454
“Hello folks”
pictures
music
...........

Add, draw, multiply, play, write, repeat, send, call, find, wait, signal, ....

Running Program

Output
98766
“Goodbye folks”
pictures
music
............
Hardware Components

Figure 1-2 Typical components of a computer system
Figure 1-4  A lab technician holds a modern microprocessor (Vadim Kolobanov/Shutterstock)
Digital Devices Encode Data As Bits

- A *bit* is an electrical switch that can be on or off (0 or 1).
- A *byte* is 8 bits.
- One Gigabyte is one billion \((10^9)\) bytes

All kinds of data must be encoded as 0’s and 1’s

numbers  
words

music  
pictures  
computer instructions

software  ...
Encoding Integers

Can store 256 different integers in a byte

1   00000001
2   00000010
4   00000100
8   00001000
16  00010000
32  00100000
64  01000000
128 10000000

10011101
128 + 16 + 8 + 4 + 1 = 157
Encoding Characters (letters, etc) in ASCII
One byte per character

<table>
<thead>
<tr>
<th>P</th>
<th>Y</th>
<th>T</th>
<th>H</th>
<th>O</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>89</td>
<td>84</td>
<td>72</td>
<td>79</td>
<td>78</td>
</tr>
</tbody>
</table>

01010000  01011001  01010100  01001000  01001111  01001110
Figure 1-16  A program is copied into main memory and then executed

The program is copied from secondary storage to main memory.

Disk drive

10100001 10111000 10011110

Main memory (RAM)

The CPU executes the program in main memory.

CPU
The CPU Executes Machine Language

Figure 1-17 The fetch-decode-execute cycle

1. **Fetch** the next instruction in the program.
2. **Decode** the instruction to determine which operation to perform.
3. **Execute** the instruction (perform the operation).
The Communication Gap

Programs must be written by people, and understood by people, as well as computers.
The Communication Gap

Assembly Language
Just machine language with pneumonics

Load A
Load B
Add
Store

......
A Compiler translates high level language into assembly language.
# Programming Languages vs. Human Languages

<table>
<thead>
<tr>
<th>Human Languages</th>
<th>Computer Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>The more of them you know, the easier it is to learn another.</td>
<td>The more of them you know, the easier it is to learn another.</td>
</tr>
<tr>
<td>All human languages have similar concepts (Verbs, nouns, tenses, adjectives, etc..)</td>
<td>All computer languages have similar concepts (Variables, functions, statements, control structures, loops, etc...)</td>
</tr>
<tr>
<td>A language has syntax rules and semantics</td>
<td>A language has syntax rules and semantics</td>
</tr>
<tr>
<td>It can take a long time to learn a new language --- lots of vocabulary.</td>
<td>You can learn a new programming language in months or weeks or days.</td>
</tr>
<tr>
<td>There is a lot of ambiguity ... people can usually sense the meaning from context.</td>
<td>There is never any ambiguity at all. Computers are literal! One wrong bit can crash your program!</td>
</tr>
<tr>
<td>You can leave out details. People can often infer what is missing.</td>
<td>You must supply all details. Computers are stupid!</td>
</tr>
</tbody>
</table>
Python Symbols

**Keywords:** and, as, assert, break, class, continue, def, del, elif, else, except, False, finally, for, from, global, if, import, in, is, lambda, None, nonlocal, not, or, pass, raise, return, True, try, while, with, yield

**Operators:** +, *, -, ..

**Punctuation**

**Variables:** programmer-supplied names
Software that Helps Programmers

• High Level Languages Compilers (Interpreters) – a brilliant achievement!
• Operating Systems – manage resources, e.g. memory, Input/Output (I/O) devices.
• Software packages (modules, libraries) – E.g. graphics, higher math functions
• Integrated Development Environment (IDE) – We will use IDLE in this class for writing and running Python programs
How to Think Like a Programmer and Pass this Course

Start

- Input Homework

  - Copy the Homework
    - Do the Homework
      - YES: Submit Homework → Success!
      - NO: Failure!
  - YES: Copy the Homework
    - NO: Do the Homework

Failure!