Lecture 1

8/14

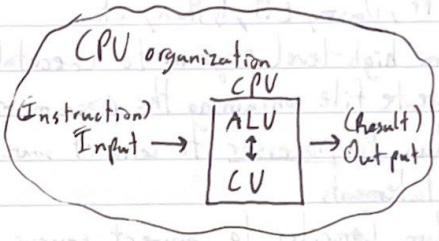
Chapter 1: Introduction to Computers and Programming

1.1 Why programming?

- Computer: Programmable machine designed to follow instructions
- Program: Instructions in computer memory to make it do something
- Programmer: Person who writes instructions to make computer perform a task

1.2 Hardware/Software

- CPU → Control Unit (Extracts & directs instruction) & Arithmetic/Logic Unit (high speed calculations/bools/y/n)
- Main Memory (RAM)
- Secondary memory
- Input devices
- Output devices



- Main memory
  - Volatile (erased when program stops/computer turns off)
  - AKA RAM
  - Bits (Binary Digit)
    - Smallest piece of memory, is a 0 or 1
  - Byte
    - 8 consecutive bits
    - Has an address
  - Addresses: Place in memory where byte is stored
    - Used to find specific bytes

Data is stored @ addresses

- Secondary Storage
  - Non-volatile (data retained when program closes/computer turned off)
  - Many types ← cloud
    - SSD, CD, flash drive, etc.

1.3 Programs & Programming Languages

- Machine Language
  - Before algorithm can be executed by computer, compiler must convert from programming language to machine language
  - Machine language instructions are in binary
    10110100011010

- Programming languages
  - Used by programmers to more easily read/write code
  - C++, Java, C#, Python, etc.
- From high-level program to executable file (.exe)
  - Create file containing the program w/ a text editor
  - Run Preprocessor to convert source file directives to source code program statements
  - Run Compiler to convert source program into machine instructions.
  - Run linker to connect hardware-specific code to machine instructions, producing an executable file

- Integrated Development Environments (IDEs)
  - Combines all tools needed to write, compile, & debug program into a single app
  - VSCode, CodeWarrior, etc.

1.4 What is a program made of

- Common elements in programming languages:
  - Key words
  - Programmer-Defined Identifiers
  - Operators
  - Punctuation
  - Syntax

- Keywords
  - AKA reserved words
  - Special meaning per language
  - int, double, return, etc. namespace
- Programmer defined identifiers
  - Names made up by programmer
  - Not part of language
  - Names of variables, functions, etc.
- Operators.
  - +, -, *, /
  - <<, >>, =
- Punctuation
  - Mark the end of a statement/seperate items in list
  - (, &, ;)
- Syntax
  - Rules of grammar that must be followed when writing a program
  - Controls use of keywords, operators, symbols, punctuation

Variables

- A named storage location in computer's memory for holding a piece of data
- Variable definitions (or declarations)
  - Variable definitions define what kind of data is to be stored.

1.5 Input, Processing, & Output

- 3 steps a program typically follows
  1. Gather input data
  2. Process the input data
  3. Display results as output

1.6 The Programming Process

1. Clearly define what the program is to do
2. Visualize the program running on the computer
3. Use design tools such as a heirarchy chart, flowcharts, or pseudocode to create a model of the program
4. Check the model for logical errors
5. Type the code, save it, & compile it
6. Correct errors found during compilation.
7. Run program with test data for input
8. Correct errors found while running program
9. Validate program results

1.7 Procedural & Object-Oriented Programming

- Procedural Programming
  - Focuses on the process. Functions are written to process data.
- Object-Oriented Programming
  - Focus on the objects, which contain data & the means to manipulate the data.