Introduction to Computers and Programming

Some definitions
- Algorithm:
  - A procedure for solving a problem
  - A sequence of discrete steps that defines such a procedure, or method
- Computer Science:
  - Finding solutions to problems with the aid of computers
  - The study of algorithms

Hardware
- Physical parts of a computer system
- Major components include:
  - CPU
  - Memory
  - Peripheral devices
CPU

- CPU consists of the control unit, the ALU, and registers
  - registers: high-speed memory cells
  - ALU: arithmetic logic unit; performs calculations
  - control unit: copies data & instructions from memory, decodes instructions and follows them

Main memory

- Stores data & instructions in bit form
  - data: information to be processed
  - instructions: commands for computer to follow
  - bits: binary digits; two states represented — as 1 or 0
- Memory is organized into 8-bit groups called bytes
- Each byte has its own unique address

Memory representation

- Each instruction or piece of data must be stored in one or more bytes of memory
- The arrangement of 1s and 0s in each 8-bit byte can be interpreted as code representing a single instruction or data item
- Many data items and instructions require more than one byte of memory; we’ll see why later
Memory size

- We typically discuss memory in increments of millions (megabytes) or billions (gigabytes)
- In fact, since the fundamental building-block of memory is the byte, which is in turn based on the bit, the actual size of what we call a megabyte isn’t really a million bytes, but the closest power of 2: $2^{20}$, or 1,048,576
- Likewise, a gigabyte is actually $2^{30}$ bytes, which is 1,073,741,824

Software

- Consists of programs that can run on a computer
  - Applications programs
  - System programs
- A program is a set of instructions
- Data are the raw material: letters and numbers, for example

Character data representation

- Bit patterns are used to represent characters and symbols
- ASCII is a code system that matches 8-bit patterns with natural language symbols, such as letters, numbers, and punctuation marks; a total of 256 characters ($2^8$) can be represented
- Unicode, a 16-bit system, is used in Java: this provides support for multiple languages and alphabets
**Integer data representation**

- Integers, or whole numbers, are represented using the binary, or base 2, numbering system.
- As in the more familiar base 10, or decimal notation, a 0 digit represents a placeholder; other digits should be multiplied by the corresponding power the base, starting with the 0th power for the rightmost digit, the 1st power for the next one, the 2nd for the next, etc.
- The sum of the digits multiplied by their respective base powers is the value of the number.

---

**Decimal example**

- The number 12,486 can be read (left to right) as:
  \[
  (1 \times 10^4) + (2 \times 10^3) + (4 \times 10^2) + (8 \times 10^1) + (6 \times 10^0) \
  \]
  
  \[
  10000 + 2000 + 400 + 80 + 6 \
  \]
- For binary numbers, the principle works the same, except that only 1s and 0s are used, and the digit at each place is multiplied by 2 instead of 10; the sum is still the value (in base 10) of the number.

---

**Binary example**

- The number 11000110, can be read as:
  \[
  (1 \times 2^5) + (1 \times 2^4) + 0 + 0 + 0 + (1 \times 2^1) + 0 + 0 = \
  128 + 64 + 4 + 2 = 198_{10} \
  \]
- Using binary representation requires several more digits than does decimal.
- The largest number that can be represented in 8 bits is 11111111, or 255_{10}.
- If one bit is reserved for the sign (positive or negative), than the magnitude of the number is restricted to \( \pm 2^7-1 \) (127_{10}).
**Integer representation**

- The number of bits used determines the magnitude of the numbers that can be stored
- Signed numbers have more restricted magnitude than unsigned numbers

**Representing real numbers**

- Like integers and characters, real numbers are stored in binary form
- Unlike integers, real numbers are represented using scientific notation:
  - in decimal, the number 12345 can be represented as $1.2345 \times 10^4$, abbreviated $1.2345e4$
  - similarly, .00025 is represented as $2.5e-4$

**Representing real numbers**

- In computers, binary notation is used
- Since binary numbers are composed of 0s and 1s, any number can be represented with a whole part of 1; so it isn’t necessary to store a number’s whole part, as it is always 1
- Therefore, several bits can be set aside to represent the fractional part of the mantissa, and several more bits can be used to represent the exponent of the radix, 2
Representing real numbers

- Both the mantissa (its fractional part) and the exponent are stored; total number of bits is apportioned between the two parts
- Both components have a sign bit
- Number of bits used restricts both the magnitude and the precision of the number to be represented; real numbers are always approximations

Representation of Instructions

- Instruction set: set of operations a particular processor can perform
  - Specific to each hardware platform
  - Encoded as binary digits
  - Programming language: language in which a program is written; consists of a set of symbols and the rules for their use

Low-level languages

- Low-level languages operate at a low level of abstraction: that is, individual symbols have a simple, well-defined, specific meaning
- Low-level languages operate at or close to the level of the computer’s instruction set; that is, the set of commands a processor is wired to respond to
Low-level languages

- Specific and simple instructions, non-abstract
- Platform-specific and non-portable
- Difficult for humans to read and write
- Include machine and assembly languages
- Programs tend to be long, as each instruction is extremely specific

High-level languages

- High-level languages exist at a higher level of abstraction; individual instructions are often equivalent to several low-level instructions
- High-level languages are closer to the natural languages humans use to communicate

High-level languages

- High-level language programs require translation to machine language before they can be executed by a computer
- As long as you have the software to do the translation, you can run a high-level language program on any kind of computer - thus, high-level languages are said to portable and platform non-specific
High-level languages

- Because the language is more abstract, a high-level language program tends to be much shorter than its low-level counterpart
- Most well-known languages are high-level languages - these include C, C++, Java, FORTRAN, BASIC, COBOL, Pascal, python, Ada, etc.

High-level languages

- Abstract
- Portable
- Relatively easy for humans to read and write
- Include most “named” languages
- Programs relatively brief
- Require translation to make executable

Programming Language Translation

- Interpreters: perform line-by-line translation and execution of a program
- Compilers: translate entire program into machine language prior to execution
- Java is unique: uses both methods