

CMSC 131: Chapter 1 (Supplement)

Introduction to Computer and Programming Systems

Computer Systems Overview

Computer systems:

Hardware -

Software -

Hardware Components:

CPU -

Main Memory - (or Random Access Memory, RAM).

Secondary Memory -

Input and Output Devices (I/O) -

Computer Systems Overview

Main memory:

- Data are encoded as **binary** numbers. Binary digit (0 or 1) is called a **bit**. 8 bits forms one **byte**. A **word** is typically 4 bytes, or 32 bits.
- Main memory is subdivided into units, called **cells**.
- The location of a memory cell is called its **address**.
- The smallest addressable unit is typically one byte, but consecutive bytes can be used to store longer data items.
- With k bits you can store up to 2^k different values.

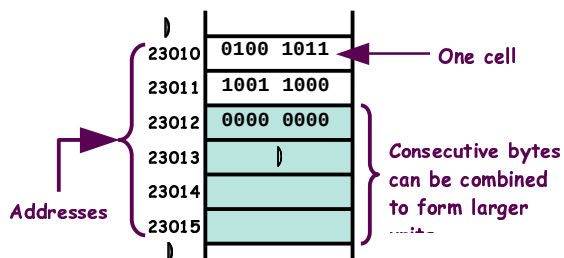
(byte) $2^8 = 256$

(word) $2^{32} \approx 4$ billion

Kilobyte $2^{10} = 1024$

Megabyte $2^{20} \approx 1$ million

Gigabyte $2^{30} \approx 1$ billion



Computer Systems Overview

Program Execution:

- Programs are **stored** permanently in secondary memory (e.g. on hard disk). Instructions are encoded as binary numbers (**machine code**).
- When a program is to be run, it is copied (**loaded**) into main memory, and the CPU **executes** the instructions of the program.
- Program may **read** data from secondary memory and input devices.
- Through **flow-control** (if-then-else, loops) the program can control the order in which program instructions are executed.
- Program data (**variables**) are stored in main memory.
- Results are **written** to output devices or to secondary memory.

Computer Systems Overview

Major Software Categories

Operating System (OS): Manages the computer's resources.

Process management:

Memory management:

I/O, Window System, and Network Control:

Enforces Security:

Applications Software:

Any software other than the operating system.

Programming Languages

Types of programming languages

Machine code -

Assembly language -

High-level languages -

1950's-early 60's: **Fortran** (scientific), **Cobol** (business)

late 60's: **Algol**

70's: **Pascal**

80's: **C**

C++

90's: **Java**

00's: **C#**

Modern Program Development

Old way:

Text editor (vi, emacs):

Compiler:

Debugger:

Modern way: Integrated Development Environment (IDE).

Installing Eclipse

Go to course webpage: Resources → Eclipse Tutorial

<http://www.cs.umd.edu/class/spring2005/cmsc131/EclipseTutorial/index.html>

- Download Java.
- Download Eclipse.
- Install and Initialize Eclipse.
- Download Plug-ins (DrJava) and restart Eclipse.

Create a Project

Create your first Java Project: File → New → Project

Provide Project Name

Type project name into window ... and type "No" to next question.

Dr. Java Eclipse Platform

Here's what you see next. The standard view that you will see throughout the semester ...

Saving, Compiling, and Running Java Code

- First add a new file (class) to a project: Read about this in tutorial.
- Check clock that says "public static void main(..)" which automatically creates this:

```
public static void main(String[] args) {  
    System.out.println("Hello world!"); ←You insert this!  
}
```

- In Eclipse saving a file compiles the file for us: File→Save.
- Next select Run → Run As → Java Application. (See what happens.)
- Exiting Eclipse:
 - Hit X in upper right corner
 - Select File □ Exit
 - Press ALT-F4

Programming Errors

Syntax:

Semantics: The program fails to satisfy its operational specifications.

Run-time errors:

Logic errors:

Debugging: