Introduction to Java Programming
Part 1: Computer and Program

Course: CPSC 1150

Instructor: Dr. Bita Shadgar

Lecture 1

Learning Outcomes

- Identify different part of a computer
- Identify and recognize the relation between hardware and software of computer
- Define program, different level of programming languages
- Identify the role of operating system as an interface between application software and hardware
- Justify the role of high-level language

The Computer Discipline

Required skills:

- Algorithmic thinking
- Design
- Representation
- Programming

Question: Mathematics vs. science vs. engineering where does computing live?

Answer: Theory vs. experimentation vs. design

Questions about CPSC 1150

Who? Langara students who wish to learn how to solve complex real world problems using computer

Questions about CPSC 1150

Who? Langara students who wish to learn how to solve complex real world problems using computer

What? A first course in how to solve problems by programming, using the Java programming language.

Questions about CPSC 1150

Who? Langara students who wish to learn how to solve complex real world problems using computer

What? A first course in how to solve problems by programming, using the Java programming language.

Where? It looks like you already figured this out!

Questions about CPSC 1150

Who? Langara students who wish to learn how to solve complex real world problems using computer

What? A first course in how to solve problems by programming, using the Java programming language.

Where? It looks like you already figured this out!

When? Check out the course outline on D2L for this (and other) important information: https://d2l.langara.bc.ca/d2l/home

Questions about CPSC 1150

Who? Langara students who wish to learn how to solve complex real world problems using computer

What? A first course in how to solve problems by programming, using the Java programming language.

Where? It looks like you already figured this out!

When? Check out the course outline on D2L for this (and other) important information: https://d2l.langara.bc.ca/d2l/home

Why? We will talk about this semester

What is programming?

Definition

Programming is creating software, or programs

A **program** is a set of instructions telling a computer what to do

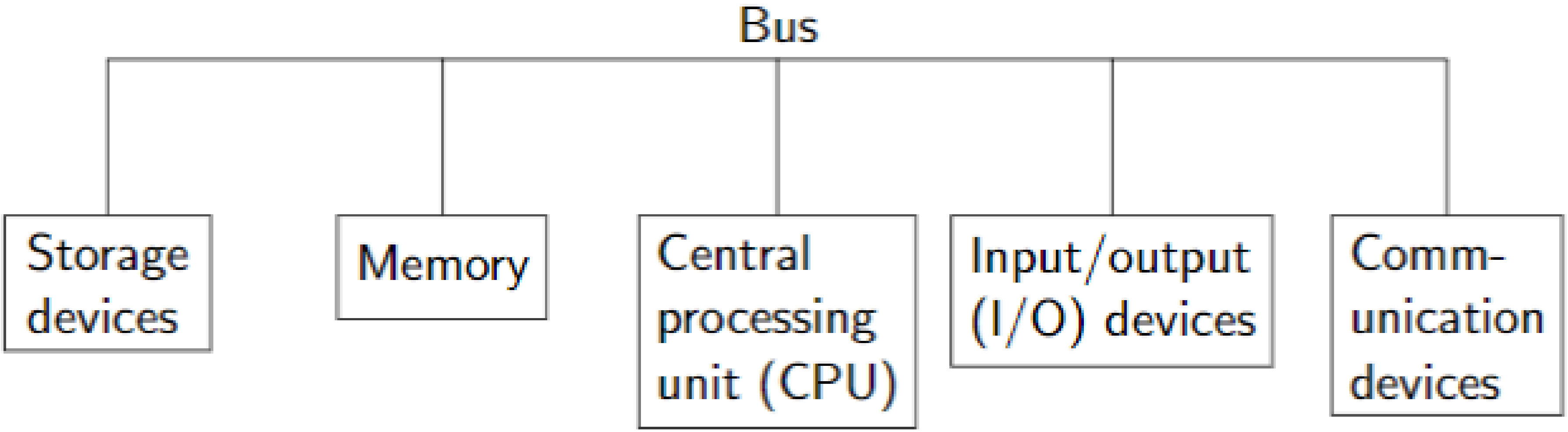
- Todays, computers are found every where!
- Computers have two types of components:
  - Software: Invisible collection of programs.

    What we will discuss for the rest of the course.
  - Hardware: Physical and tangible parts.

    Overview of this today.

Computer Hardware

- As programmers, understanding hardware helps us to know what our programs are really doing



- Components communicate via the bus.
- We'll learn a little about each type of hardware component

Central Processing Unit

- Where the magic happens : The ‘Brain’ of a computer
- Physically made from a small silicon chip with millions of **transistors** (electronic switches)
- **Multiple cores** in a CPU can read and execute instructions independently
- CPU retrieves instructions from memory and executes them

CPU Components

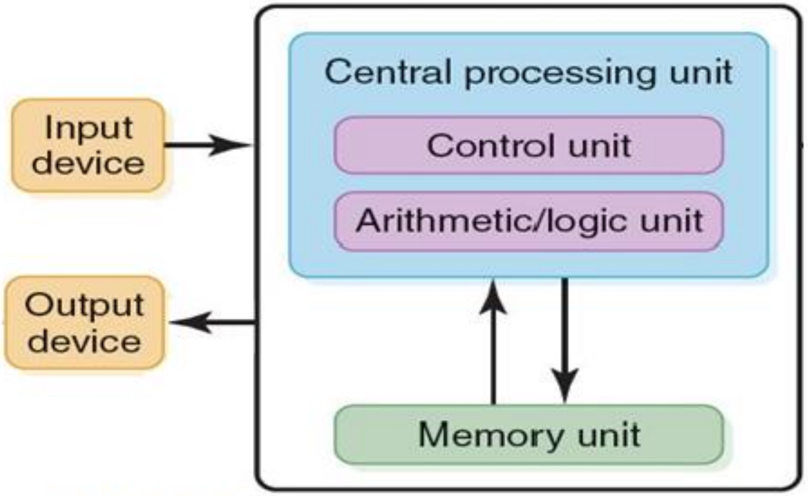


FIGURE 5.1 The von Neumann architecture.

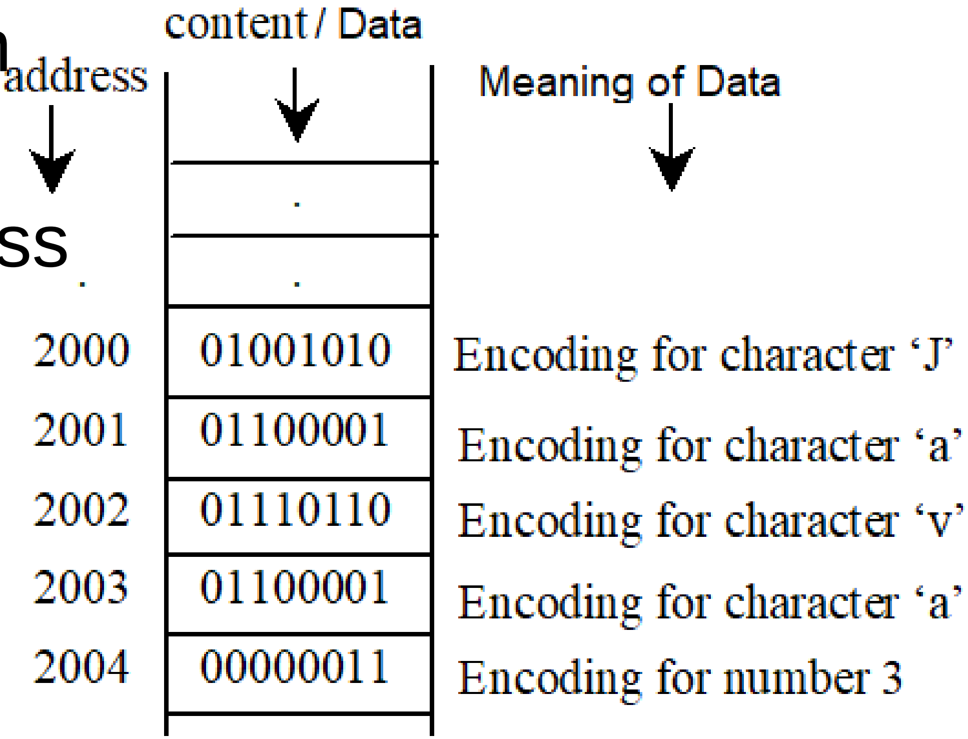
- **Clock** - emits regular pulses
  - Clock speed is measured in Hertz (Hz) = pulses per second
  - Current CPUs can run at 3 GHz, or about 3 billion pulses per second
- **Control unit** - executes instructions each clock cycle
- **ALU** - arithmetic/logic unit

Information in a computer

- Computer can be thought of as a collection of switches
  - Storing data/programs just turns certain switches on and off
- **Bits** (binary digits) represent the state of switches
  - The \`on' position is represented as a 1
  - The \`off ' position is represented as a 0
- Minimum unit of storage is one **byte** = 8 bits
- An encoding scheme (like ASCII) is used to translate characters, numbers, and symbols into byte
  - Example : The byte 01100001 represents the character \`a' under the ASCII encoding scheme

Memory

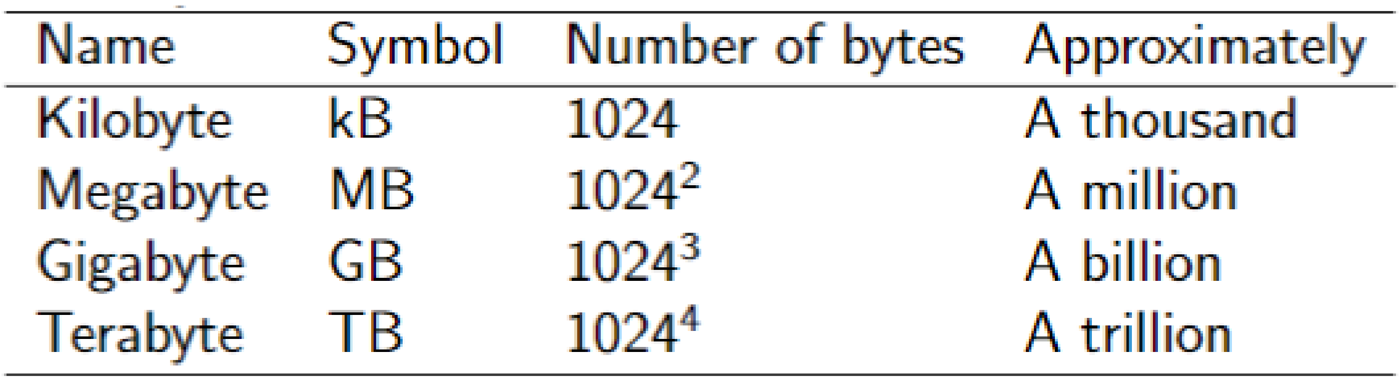
- Short-term storage
- Called random access memory (RAM) since it doesn't have to be accessed in order
- Memory is an ordered in sequence of bytes
- Each byte has an address
- Bytes never get erased, just overwritten



address	content / Data	Meaning of Data
	.	
.	.	
2000	01001010	Encoding for character ‘J’
2001	01100001	Encoding for character ‘a’
2002	01110110	Encoding for character ‘v’
2003	01100001	Encoding for character ‘a’
2004	00000011	Encoding for number 3

Memory (cont.)

- A program and associated data need to be in memory to be executed
- Physically, memory is built on silicon chips with millions of transistors, like a CPU
- Everything in memory is reset when power is turned off
- Memory size is measured in:



Name	Symbol	Number of bytes	Approximately
Kilobyte	kB	1024	A thousand
Megabyte	MB	1024^2	A million
Gigabyte	GB	1024^3	A billion
Terabyte	TB	1024^4	A trillion

- Today, memory on personal computers is usually between 6-8 GB

Storage devices

- Long-term memory
- Slower to access than RAM
- Different types of media, and each one has a drive to read from and write to it:
  - Magnetic disk drive - main chunk of storage on a computer
  - Optical disk drive - reads/writes CDs and DVDs
  - USB flash drive - Universal Serial Bus allows many I/O devices to connect to a computer, including a flash drive

Input/output (I/O) devices

Hardware that allows the user to interact with the computer

- Examples of input devices
  - Keyboard
  - Mouse/touchpad
  - Touchscreen
- Examples of output devices
  - Monitor
  - Printer
  - Speaker

Communication devices

Hardware that allows the computer to communicate with a network

- Examples
  - Dial-up/cable modem
  - Digital subscriber line (DSL)
  - Network adapter

Levels of programming languages

- Low level
  - Understandable to a computer
    - e.g. Machine language
- Mid-level
    - e.g. Assembly language
- High level
  - Understandable to a human
    - e.g. High-level languages such as Java and Python

Machine language

- The code that computers understand and can execute
  - Example

    Machine code as follows: 111001011 00000110 may contain an instruction to add 6 to a stored number
- Sequences of bytes represent primitive built-in instructions, that differ from machine to machine
- Extremely difficult for humans to create, edit and maintain this kind of code

Assembly language

- An instruction in assembly is basically shorthand for a machine code instruction
  - Example

    An assembly instruction to add 6 to a stored number might look like this: ADDX 06
- Written in a way that humans can (tediously) understand and manipulate
- Assembly source code gets passed through a program called an assembler which spits out equivalent machine code

High-level languages

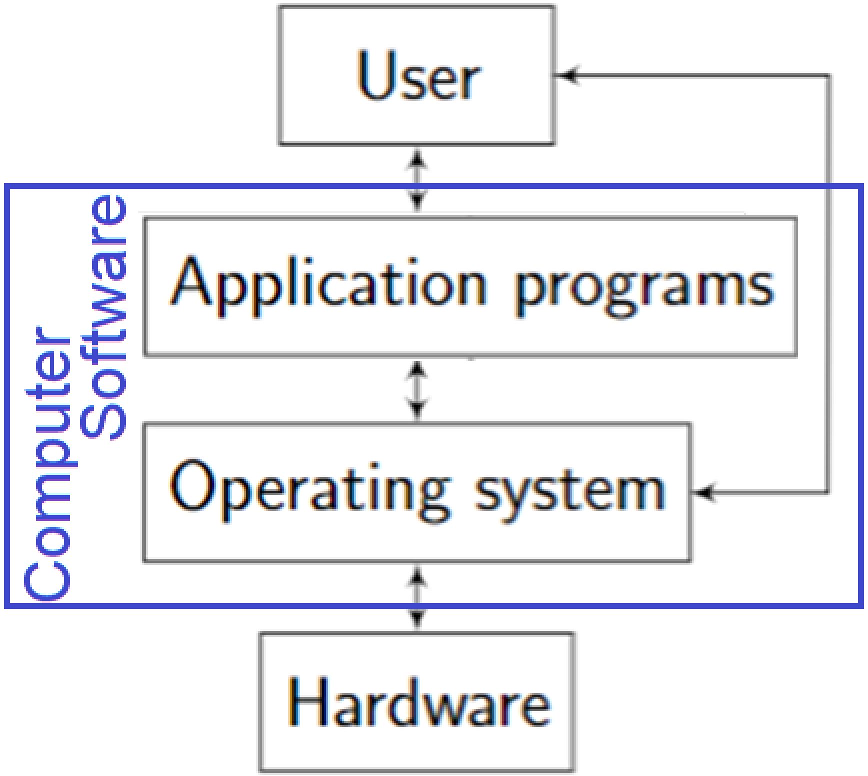
- Most similar to natural languages (i.e., English)
- Platform (machine) independent
- A chunk of code in a high-level language is called source code
- Source code must be compiled or interpreted before it is run on a computer
  - Compiled: Source code -> Compiler -> Machine code -> CPU
  - Interpreted: Source code -> Interpreter -> CPU

Example

- A statement in a high level language to add 6 to a stored number might look like this: x = x + 6;

Computer Software

- System Software
  - Examples:
    - Operating System
    - Device Drivers
    - Compilers and more
- Application Software
  - Examples:
    - Science, Business, Graphics, Games



Operating system (OS)

- Most important program
- Manages and controls a computer
- All application programs must run on top of an OS

Example

- Some popular operating systems are Windows, Mac OS, and Linux

Main functions of the OS

- Controls system activities
  - I/O and access to storage go through the OS
  - OS ensures programs don't interfere with each other
  - Security is managed by the OS

- Allocates resources to programs
  - CPU time, memory space, disk use, I/O, etc.

- Schedules operations to make efficient use of resources
  - Multiprogramming, Multithreading, Multiprocessing

More Practice

- List the key terms in the lecture
- Define each key term
- Name the different parts of a computer
- Explain how a computer works