CSC 1051 – Data Structures and Algorithms I

Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Mendel 162C

Course website:
www.csc.villanova.edu/~map/1051/
Last time:

• Go over syllabus/course information
  – www.csc.villanova.edu/~map/1051

• Introduction to the course – reverse history of computing

• Take the online survey
Today:

• Data representation
• Software
• A glimpse of Java
Reverse History of computing

Dig deeper into what we already know, travel backwards…

1. What we see now all around us – a connected world of computing

2. Focus on a single “traditional” computer

3. Dig deeper – data and processing
CPU and Main Memory

- Central Processing Unit
- Main Memory
- Chip that executes program commands
- Primary storage area for programs and data that are in active use
- Synonymous with RAM
CPU and Main Memory

- Central Processing Unit
- Main Memory

Primary storage area for programs and data that are in active use

Synonymous with RAM

Chip that executes program commands

Historic note: Von Neuman architecture
John Von Neuman, USA 1945
Main memory is divided into many memory locations (or *cells*).

Each memory cell has a numeric *address*, which uniquely identifies it.
Storing Information

Each memory cell stores a set number of bits (usually 8 bits, or one byte)

Large values are stored in consecutive memory locations
Storage Capacity

- Every memory device has a *storage capacity*, indicating the number of bytes it can hold.

- Capacities are expressed in various units:

<table>
<thead>
<tr>
<th>Unit</th>
<th>Symbol</th>
<th>Number of Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>kilobyte</td>
<td>KB</td>
<td>$2^{10} = 1024$</td>
</tr>
<tr>
<td>megabyte</td>
<td>MB</td>
<td>$2^{20}$ (over one million)</td>
</tr>
<tr>
<td>gigabyte</td>
<td>GB</td>
<td>$2^{30}$ (over one billion)</td>
</tr>
<tr>
<td>terabyte</td>
<td>TB</td>
<td>$2^{40}$ (over one trillion)</td>
</tr>
<tr>
<td>petabyte</td>
<td>PB</td>
<td>$2^{50}$ (a whole bunch)</td>
</tr>
</tbody>
</table>
The Central Processing Unit

- A CPU is on a chip called a *microprocessor*
- It continuously follows the *fetch-decode-execute cycle*:

  - **fetch**: Retrieve an instruction from main memory
  - **decode**: Determine what the instruction is
  - **execute**: Carry out the instruction
The Central Processing Unit

- A CPU is on a chip called a *microprocessor*
- It continuously follows the *fetch-decode-execute cycle*:
  - **fetch**: Retrieve an instruction from main memory
  - **decode**: Determine what the instruction is
  - **execute**: Carry out the instruction

*System clock controls speed, measured in gigahertz (GHz)*
The Central Processing Unit

- **Arithmetic / Logic Unit**: Performs calculations and makes decisions.
- **Control Unit**: Coordinates processing (system clock, decoding, etc).
- **Registers**: Small, very fast memory.
Automatic control of computation

- The concept of a machine that can follow a series of steps - a “program”
- Some early steps:
  - Jacquard loom (1801)
  - Babbage's Difference engine and Analytical engine (1822)
  - Holerith's census machine (1890)
- Stored program and the fetch/decode/execute cycle (John von Neumann, 1945)
- ENIAC - first fully electronic digital computer (Eckert and Mauchley, 1946)
Analog vs. Digital Data

- **Analog**
  - continuous, in direct proportion to the data represented
  - music on a record album - a needle rides on ridges in the grooves that are directly proportional to the voltages sent to the speaker

- **Digital**
  - information is broken down into pieces, and each piece is represented separately
  - *sampling* – record discrete values of the analog representation
Representing Text Digitally

• For example, every character is stored as a number, including spaces, digits, and punctuation

• Corresponding upper and lower case letters are separate characters

Hi, Heather.

72 105 44 32 72 101 97 116 104 101 114 46

01100001 binary

ASCII / UNICODE

CSC 1051 M.A. Papalaskari, Villanova University
Binary Numbers

- Number system consisting of 1’s & 0’s
- Simplest way to represent digital information
- Modern computers use binary numbers internally

A binary digit is called a *bit* - *binary digit*
Eight bits together is called a *byte*
Bit Permutations

<table>
<thead>
<tr>
<th>1 bit</th>
<th>2 bits</th>
<th>3 bits</th>
<th>4 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>000</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>01</td>
<td>001</td>
<td>0001</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>010</td>
<td>0010</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>011</td>
<td>0011</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td></td>
<td>0100</td>
</tr>
<tr>
<td></td>
<td>101</td>
<td></td>
<td>0101</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td></td>
<td>0110</td>
</tr>
<tr>
<td></td>
<td>111</td>
<td></td>
<td>0111</td>
</tr>
</tbody>
</table>

Each additional bit doubles the number of possible permutations
Bit Permutations

- Each permutation can represent a particular item
- There are $2^N$ permutations of N bits
- Therefore, N bits are needed to represent $2^N$ unique items

How many items can be represented by

<table>
<thead>
<tr>
<th>Number of Bits</th>
<th>Calculation</th>
<th>Number of Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 bit</td>
<td>$2^1 = 2$</td>
<td>2 items</td>
</tr>
<tr>
<td>2 bits</td>
<td>$2^2 = 4$</td>
<td>4 items</td>
</tr>
<tr>
<td>3 bits</td>
<td>$2^3 = 8$</td>
<td>8 items</td>
</tr>
<tr>
<td>4 bits</td>
<td>$2^4 = 16$</td>
<td>16 items</td>
</tr>
<tr>
<td>5 bits</td>
<td>$2^5 = 32$</td>
<td>32 items</td>
</tr>
</tbody>
</table>
Quick Check

How many bits would you need to represent each of the 50 United States using a unique permutation of bits?
Quick Check

How many bits would you need to represent each of the 50 United States using a unique permutation of bits?

Five bits wouldn't be enough, because $2^5$ is 32.

**Six bits** would give us 64 permutations, and some wouldn't be used.

<table>
<thead>
<tr>
<th>Bits</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>Alabama</td>
</tr>
<tr>
<td>000001</td>
<td>Alaska</td>
</tr>
<tr>
<td>000010</td>
<td>Arizona</td>
</tr>
<tr>
<td>000011</td>
<td>Arkansas</td>
</tr>
<tr>
<td>000100</td>
<td>California</td>
</tr>
<tr>
<td>000101</td>
<td>Colorado</td>
</tr>
<tr>
<td>etc.</td>
<td></td>
</tr>
</tbody>
</table>
Binary Representation of Information

- Computers store all information digitally, in binary:
  - numbers
  - text
  - graphics and images
  - audio
  - video
  - program instructions
Mechanization of arithmetic

• Historic note – the creation of various special purpose calculators
  – Abacus (2400 BC)
  – Number systems (Babylonian, Greek, Roman, Arabic 1000 BC = 600 AD)
  – Stonehenge (1900-1600 BC)
  – Napier's bones (1600, a precursor of the slide rule)
  – Pascal's adder (1642)
  – Leibniz's calculator (1670s)
  – modern calculators
Today: Representing and processing bits

- Electronic circuits: high/low voltage
- Magnetic devices (e.g., hard drive): positive/negative
- Optical devices (e.g., DVD): light reflected/not reflected due to microscopic grooves
Reminder: Storage Capacity

- Every memory device has a storage capacity, indicating the number of bytes it can hold.
- Capacities are expressed in various units:

<table>
<thead>
<tr>
<th>Unit</th>
<th>Symbol</th>
<th>Number of Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>kilobyte</td>
<td>KB</td>
<td>$2^{10} = 1024$</td>
</tr>
<tr>
<td>megabyte</td>
<td>MB</td>
<td>$2^{20}$ (over one million)</td>
</tr>
<tr>
<td>gigabyte</td>
<td>GB</td>
<td>$2^{30}$ (over one billion)</td>
</tr>
<tr>
<td>terabyte</td>
<td>TB</td>
<td>$2^{40}$ (over one trillion)</td>
</tr>
<tr>
<td>petabyte</td>
<td>PB</td>
<td>$2^{50}$ (a whole bunch)</td>
</tr>
</tbody>
</table>
Memory characteristics

• **Volatile** - stored information is lost if the electric power is removed

• **Direct access** or **Random access** – information can be reached directly (as opposed to sequentially as in the case of magnetic tape)

• **Read/Write** – information can be overwritten (as opposed to read-only devices – ROM)
## Random Access Memory Devices

<table>
<thead>
<tr>
<th>Volatile</th>
<th>Non-volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td>fastest</td>
<td>ROM chip</td>
</tr>
<tr>
<td>CPU registers</td>
<td>ROM chip</td>
</tr>
<tr>
<td>Cache memory</td>
<td>ROM chip</td>
</tr>
<tr>
<td>fast</td>
<td>USB flash drive</td>
</tr>
<tr>
<td>main memory</td>
<td>Hard disks</td>
</tr>
<tr>
<td>(Also called Random Access Memory -- RAM)</td>
<td>CD-ROM</td>
</tr>
<tr>
<td>slow</td>
<td>DVD</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Random Access Memory Devices

<table>
<thead>
<tr>
<th>Volatile</th>
<th>Non-volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td>fastest CPU registers Cache memory</td>
<td>ROM chip</td>
</tr>
<tr>
<td>fast main memory (Also called Random Access Memory -- RAM)</td>
<td>ROM chip</td>
</tr>
<tr>
<td>slow</td>
<td>USB flash drive Hard disks CD-ROM DVD</td>
</tr>
</tbody>
</table>
## Random Access Memory Devices

<table>
<thead>
<tr>
<th>Volatile</th>
<th>Non-volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td>fastest</td>
<td></td>
</tr>
<tr>
<td>CPU registers</td>
<td>ROM chip</td>
</tr>
<tr>
<td>Cache memory</td>
<td></td>
</tr>
<tr>
<td>main memory (Also called Random Access Memory -- RAM)</td>
<td>ROM chip</td>
</tr>
<tr>
<td>fast</td>
<td></td>
</tr>
<tr>
<td></td>
<td>USB flash drive</td>
</tr>
<tr>
<td>slow</td>
<td></td>
</tr>
<tr>
<td></td>
<td>magnetic Hard disks</td>
</tr>
<tr>
<td></td>
<td>CD-ROM DVD</td>
</tr>
</tbody>
</table>

CSC 1051 M.A. Papalaskari, Villanova University
## Random Access Memory Devices

<table>
<thead>
<tr>
<th></th>
<th>Volatile</th>
<th>Non-volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>fastest</strong></td>
<td>CPU registers</td>
<td>ROM chip</td>
</tr>
<tr>
<td></td>
<td>Cache memory</td>
<td></td>
</tr>
<tr>
<td><strong>fast</strong></td>
<td>main memory</td>
<td>ROM chip</td>
</tr>
<tr>
<td></td>
<td>(Also called Random Access Memory -- RAM)</td>
<td></td>
</tr>
<tr>
<td><strong>slow</strong></td>
<td></td>
<td>USB flash drive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hard disks</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>optical</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CD-ROM LCD</td>
</tr>
</tbody>
</table>

CSC 1051 M.A. Papalaskari, Villanova University
Compact Discs

- A CD-ROM is portable read-only memory
- A microscopic pit on a CD represents a binary 1 and a smooth area represents a binary 0
- A low-intensity laser reflects strongly from a smooth area and weakly from a pit
- A CD-Recordable (CD-R) drive can be used to write information to a CD once
- A CD-Rewritable (CD-RW) can be erased and reused
- The speed of a CD drive indicates how fast (max) it can read and write information to a CD
A DVD is the same physical size as a CD, but can store much more information.

The format of a DVD stores more bits per square inch.

A CD can store 650 MB, while a standard DVD can store 4.7 GB.
  - A double sided DVD can store 9.4 GB.
  - Other advanced techniques can bring the capacity up to 17.0 GB.

Like CDs, there are DVD-R and DVD-RW discs.
**RAM vs. ROM**

- **RAM** - Random Access Memory
  - synonymous with main memory:
    - fast
    - read/write
    - volatile
    - random access

- **ROM** - Read-Only Memory
  - ROM typically holds the firmware, e.g., BIOS
    - fast (except in CD-ROM)
    - read only
    - non-volatile
    - random access
Modern computer

Great human developments that gave rise to the modern computer:

- Number systems and other encodings – data representation
- Mechanization of arithmetic – the concepts of algorithms and computation
- Automatic control of computation – a “program” to control operations (fetch/decode/execute cycle and the stored program concept)
Hardware and Software

- **Hardware**
  - the physical, tangible parts of a computer
  - keyboard, monitor, disks, wires, chips, etc.

- **Software**
  - programs and data
  - a *program* is a series of instructions

- A computer requires both hardware and software

- Each is essentially useless without the other
Software Categories

• Operating System
  – controls all machine activities
  – provides the user interface to the computer
  – manages resources such as the CPU and memory
  – Windows, Mac OS, Unix, Linux,

• Application program
  – generic term for any other kind of software
  – word processors, missile control systems, games

• Most operating systems and application programs have a *graphical user interface* (GUI)
Communicating with a Computer

- Programming languages
  - Bridge the gap between human thought and
  - Computer binary circuitry

- Programming language:
  - A series of specifically defined commands
  - Given by human programmers
  - To give directions to the digital computers
Translation Needed

- Special program to translate into binary
- Programmer writes – **Source code**
- Translation produces the binary equivalent – **Object code**
- The translator is an assembler, compiler, or an interpreter
  - Takes in the source code
  - Yields computer understandable instructions
Object-Oriented

• Java is an Object Oriented Language

• Object-Oriented Languages
  – Expresses a problem as a series of objects
    • Example student
  – Once an object / class is created, we can put it on the shelf and use it over and over again
  – Reuse is what makes Java such a rich language
  – Many Classes already exist and are in a library for our use
Java

• A *programming language* specifies the words and symbols that we can use to write a program.

• A programming language employs a set of rules that dictate how the words and symbols can be put together to form valid *program statements*.

• The Java programming language was created by Sun Microsystems, Inc.

• It was introduced in 1995 and its popularity has grown quickly since.
Java Program Structure

• In the Java programming language:
  – A program is made up of one or more classes
  – A class contains one or more methods
  – A method contains program statements

• These terms will be explored in detail throughout the course

• A Java application always contains a method called main

• See Lincoln.java
public class Lincoln
{
    // Prints a presidential quote.
    public static void main (String[] args)
    {
        System.out.println("A quote by Abraham Lincoln:");
        System.out.println("Whatever you are, be a good one.");
    }
}
public class Lincoln {

    // Prints a presidential quote.
    public static void main (String[] args) {
        System.out.println ("A quote by Abraham Lincoln:");
        System.out.println ("Whatever you are, be a good one.");
    }
}
Java Program Structure

// comments about the class
public class MyProgram
{
    // class header
    
    // class body
    Comments can be placed almost anywhere
}

CSC 1051 M.A. Papalaskari, Villanova University
Java Program Structure

// comments about the class
public class MyProgram {

    // comments about the method
    public static void main (String[] args) {
        method body
    }

}
Comments

• Comments in a program are called *inline documentation*

• They should be included to explain the purpose of the program and describe processing steps

• They do not affect how a program works

• Java comments can take three forms:
  
  ```
  // Basic this comment runs to the end of the line
  
  /* Basic this comment runs to the terminating symbol, even across line breaks */
  
  /** this is a javadoc comment */
  ```
Identifiers

- **Identifiers** are the words a programmer uses in a program
- An identifier can be made up of letters, digits, the underscore character ( _ ), and the dollar sign
- Identifiers cannot begin with a digit
- Java is *case sensitive* - Total, total, and TOTAL are different identifiers
- By convention, programmers use different case styles for different types of identifiers, such as
  - *title case* for class names - Lincoln
  - *upper case* for constants - MAXIMUM
Identifiers

• Sometimes we choose identifiers ourselves when writing a program (such as Lincoln)

• Sometimes we are using another programmer's code, so we use the identifiers that he or she chose (such as println)

• Often we use special identifiers called reserved words that already have a predefined meaning in the language

• A reserved word cannot be used in any other way
## Reserved Words

- The Java reserved words:

<table>
<thead>
<tr>
<th>reserved词</th>
<th>reserved词</th>
<th>reserved词</th>
<th>reserved词</th>
</tr>
</thead>
<tbody>
<tr>
<td>abstract</td>
<td>else</td>
<td>interface</td>
<td>switch</td>
</tr>
<tr>
<td>assert</td>
<td>enum</td>
<td>long</td>
<td>synchronized</td>
</tr>
<tr>
<td>boolean</td>
<td>extends</td>
<td>native</td>
<td>this</td>
</tr>
<tr>
<td>break</td>
<td>false</td>
<td>new</td>
<td>throw</td>
</tr>
<tr>
<td>byte</td>
<td>final</td>
<td>null</td>
<td>throws</td>
</tr>
<tr>
<td>case</td>
<td>finally</td>
<td>package</td>
<td>transient</td>
</tr>
<tr>
<td>catch</td>
<td>float</td>
<td>private</td>
<td>true</td>
</tr>
<tr>
<td>char</td>
<td>for</td>
<td>protected</td>
<td>try</td>
</tr>
<tr>
<td>class</td>
<td>goto</td>
<td>public</td>
<td>void</td>
</tr>
<tr>
<td>const</td>
<td>if</td>
<td>return</td>
<td>volatile</td>
</tr>
<tr>
<td>continue</td>
<td>implements</td>
<td>short</td>
<td>while</td>
</tr>
<tr>
<td>default</td>
<td>import</td>
<td>static</td>
<td></td>
</tr>
<tr>
<td>do</td>
<td>instanceof</td>
<td>strictfp</td>
<td></td>
</tr>
<tr>
<td>double</td>
<td>int</td>
<td>super</td>
<td></td>
</tr>
</tbody>
</table>

CSC 1051 M.A. Papalaskari, Villanova University
White Space

• Spaces, blank lines, and tabs are called *white space*

• White space is used to separate words and symbols in a program

• Extra white space is ignored

• A valid Java program can be formatted many ways

• Programs should be formatted to enhance readability, using consistent indentation

• See [Lincoln2.java](Lincoln2.java), [Lincoln3.java](Lincoln3.java)
Program Development

• The mechanics of developing a program include several activities

  – writing the program in a specific programming language (such as Java)

  – translating the program into a form that the computer can execute

  – investigating and fixing various types of errors that can occur

• Software tools can be used to help with all parts of this process
Programming Languages

• Each type of CPU executes only a particular *machine language*

• A program must be translated into machine language before it can be executed

• A *compiler* is a software tool which translates *source code* into a specific target language

• Often, that target language is the machine language for a particular CPU type

• The Java approach is somewhat different
Java Translation

Java source code

Java compiler

Java bytecode

Bytecode interpreter

Bytecode compiler

Machine code
Development Environments

• There are many programs that support the development of Java software, including:
  – Sun Java Development Kit (JDK)
  – Sun NetBeans
  – IBM Eclipse
  – IntelliJ IDEA
  – Oracle JDeveloper
  – BlueJ
  – jGRASP

• Though the details of these environments differ, the basic compilation and execution process is essentially the same
Syntax and Semantics

• The syntax rules of a language define how we can put together symbols, reserved words, and identifiers to make a valid program.

• The semantics of a program statement define what that statement means (its purpose or role in a program).

• A program that is syntactically correct is not necessarily logically (semantically) correct.

• A program will always do what we tell it to do, not what we meant to tell it to do.
Errors

• A program can have three types of errors

• The compiler will find syntax errors and other basic problems (*compile-time errors*)
  – If compile-time errors exist, an executable version of the program is not created

• A problem can occur during program execution, such as trying to divide by zero, which causes a program to terminate abnormally (*run-time errors*)

• A program may run, but produce incorrect results, perhaps using an incorrect formula (*logical errors*)
Summary

• Computer processing
• Data representation
• A little bit of history
• Programming and programming languages
• An introduction to Java
Homework

• Review Chapter 1

  – **Always** do all self-review exercises when you review material

• Do Exercises EX 1.15- 1.20

• Read Section 2.1 to prepare for next class