Introduction to Algorithms and Data Structures
CSC 1051 – Algorithms and Data Structures I
Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

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

Some slides in this presentation are adapted from the slides accompanying Java Software Solutions by Lewis & Loftus

What is this course about?

- Computer Science
- Problem solving
- Algorithmic thinking
- Data representation
- Software engineering

Our textbook

Java Software Solutions
Foundations of Program Design
Seventh Edition
John Lewis
William Loftus

(6th edition is ok, but some exercise numbers may be different)
An old quote

A priest asked: “What is Fate, Master?”
And he answered:
“It is that which gives a beast of burden its reason for existence. It is that which men in former times had to bear upon their backs. It is that which has caused nations to build byways from City to City upon which carts and coaches pass, and alongside which inns have come to be built to stave off Hunger, Thirst and Weariness.”
“And that is Fate?” said the priest.
“Fate... I thought you said Freight,” responded the Master.
“That's all right,” said the priest. “I wanted to know what Freight was too.”
- Kehlog Albran

Source unknown. This quote appeared as one of the “fortunes” displayed by the fortune cookie program on old unix systems. (“fortune” was a program that ran automatically every time you logged out of a unix session and displayed a random, pithy saying.)

Reverse History of computing

Examine 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

The Internet

History: Started as a United States government project, sponsored by the Advanced Research Projects Agency (ARPA) in late 1960’s

See also: http://www.internethalloffame.org/internet-history/timeline

• 1970’s and 1980’s: ARPANET
  – wide area network
  – protocols for communication
• 1990’s: World Wide Web
  – html and web browsers

A network is two or more computers that are connected so that data and resources can be shared

A Local-Area Network (LAN) covers a small distance and a small number of computers
A Wide-Area Network (WAN) connects two or more LANs, often over long distances
The Arpanet in 1971

‘Interface Message Processor’ (IMP)

evolved into today’s routers.

The World Wide Web

The Internet Protocol (IP) determines how data are routed across network boundaries.

Each computer on the Internet has a unique IP address, such as: 204.192.116.2

Data are accessed using a Uniform Resource Locator (URL):

eg: http://www.cnn.com
ftp://java.sun.com/applets/animation.zip

• A URL specifies a protocol (http), a domain, and possibly specific documents

• Web documents are often defined using the HyperText Markup Language (HTML)

The World Wide Web

• The World Wide Web allows many different types of information to be accessed using a common interface

• Resources presented include:
  – text, graphics, video, sound, audio, executable programs

• A browser is a program which accesses network resources and presents them
  – Popular browsers: Chrome, Internet Explorer, Safari, Firefox
  – My first browser: Mosaic <3

Reverse History of computing

Examine 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
A Computer Specification

- Consider the following specification for a personal computer:
  - 3.07 GHz Intel Core i7 processor
  - 4 GB RAM
  - 750 GB Hard Disk
  - 16x Blu-ray / HD DVD-ROM & 16x DVD+R DVD Burner
  - 17” Flat Screen Video Display with 1280 x 1024 resolution
  - Network Card

Computer Architecture

Memory

Main memory is divided into many memory locations (or cells)

Each memory cell has a numeric address, which uniquely identifies it

“Random Access Memory (RAM)”

You don’t have to scan the memory sequentially – go to data directly using the address
Historic note: Von Neuman architecture
John Von Neuman, USA 1945

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

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)

Historic Notes: Automatic control of computation
- The concept of a machine that can follow a series of steps - a "program"
- Some early steps:
  - Jacquard loom (France 1801)
  - Babbage's Difference engine and Analytical engine (England 1822)
  - Holerith's census machine (USA 1890)
  - Colossus Mark I – first electronic computer to be programmable (Alan Turing, England 1944)
  - Stored program and the fetch/decode/execute cycle (John von Neumann, USA 1945)
  - ENIAC - first fully electronic digital computer (Eckert and Mauchley, University of Pennsylvania, 1946)
Jacquard Loom

This portrait of Jacquard was woven in silk on a Jacquard loom and required 24,000 punched cards to create (1839). It was only produced to order. Charles Babbage owned one of these portraits; it inspired him in using punched cards in his analytical engine. Collection of the Science Museum in London, England. (Source: Wikipedia)

Charles Babbage & Ada Lovelace

Designed the Analytical Engine

First “Programmer” for (not yet built) Analytical Engine

1945: The word “computer” changes its meaning

Captain Grace Hopper and other computers

The Electronic Numeric Integrator and Calculator (ENIAC) Programmers Betty Jean Jennings (left) and Fran Bilas (right) operate ENIAC’s main control panel at the Moore School of Electrical Engineering. (U.S. Army photo from the archives of the ARL Technical Library)

2014: Benedict Cumberbatch shows the world how cool Alan Turing was

The Imitation Game (2014)
Reverse History of computing

Examine 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

Data Representation

- Computers store all information \textit{digitally}, using \textit{binary} codes:
  - numbers
  - text
  - images
  - audio
  - video
  - program instructions

Binary Numbers

- Number system consisting of 1’s & 0’s
- Simplest way to represent digital information:
  - \textit{Electronic circuits}: high/low voltage
  - \textit{Magnetic devices} (eg hard drive): positive/negative
  - \textit{Optical devices} (eg DVD): light reflected/not reflected due to microscopic grooves

A binary digit is called a \textit{bit} - \textit{binary digit}

A \textit{byte} is a group of eight bits

Computing devices store use binary codes to represent data of \textit{all kinds}
Example: Representing Text

• Characters, including spaces, digits, and punctuation are represented by numeric codes

\texttt{Hi, Heather.}

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

The \textsc{ASCII} (American Standard Code for Information Interchange) character set uses eight bits per character, allowing for 256 unique characters.

01110100 \textsc{ASCII}

00000000 01110100 \textsc{UNICODE}

The \textsc{Unicode} character set extends \textsc{ASCII} to sixteen bits per character, allowing for 65,536 unique characters.

Example: Representing Program Instructions

Intel opcode for the instruction \texttt{JZ} (jump if zero):

01110100

Historic Note: Symbolic Representation & Mechanization of Arithmetic

• Development of number systems
  – Abacus (2400 BC)
  – Number systems (Babylonian, Greek, Roman, Arabic 1000 BC - 800 AD)

• The notion of an algorithm
  – Euclid (300 BC)
  – al-Khwārizmī (780 AD)

• Creation of special purpose calculators
  – Stonehenge (1900-1600 BC)
  – Napier’s bones (1600, a precursor of the slide rule)
  – Pascal’s adder (1642)
  – Leibniz’s calculator (1670s)
  – modern calculators
Basic human needs: **Symbolism**

:-)  ♫  π

apple  ∞  <3

### Historic note: Trends that gave rise to the modern computer

- **Symbolic representation and the mechanization of arithmetic** – the concepts of numbers, symbols, algorithms, and computation
- **Automatic control of computation** – a “program” to control operations (fetch/decode/execute cycle and the stored program concept)
- **Networks and telecommunications** – connecting computers together

= **modern computer**

---

Basic human needs: **counting & measuring**

---

**Computer Science**

A new paradigm in humanity’s search for understanding of:

- Representation & encoding
- Computation
- Problem solving
- Mechanization

**History Epilogue:** Just like Physics and other sciences branched off from philosophy during the Renaissance, so CS emerged in the 20th century from the work of philosophers and mathematicians – with the help of dedicated, visionary practitioners, experimental scientists and engineers.
Part 2 – introduction to Java

High-level programming languages

- Programmer writes **Source code**
- Translation produces the binary equivalent – **Object code**
- Translation is performed by an assembler, compiler, or interpreter (stay tuned)

Java Translation

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
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

Java Program Example

```java
package com.example;

public class Lincoln {
    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 {
    // comments about the method
    public static void main(String[] args) {
        // comments about the method
        System.out.println("Some code");
    }
}
```
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

Reserved Words

• The Java reserved words:

  abstract else interface switch
  assert enum long synchronized
  boolean extends native this
  break false new throw
  byte final null try
  case finally package transient
  catch float private true
  char for protected try
  class goto public void
  const if return volatile
  continue implements short while
  default import static strictfp
  do instanceof super
  double int
White Space (Spaces, blank lines, and tabs)

- Extra white space is ignored
- Programs should be formatted to enhance readability, using consistent indentation
- See [Lincoln2.java](Lincoln2.java), [Lincoln3.java](Lincoln3.java)

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*)

The original "bug" found in the relays of Harvard’s Mark II computer by Admiral Grace Murray Hopper’s team.

Summary

- History of computing
- Computer hardware and software overview
- An introduction to Java:
  - a first program
  - identifiers
  - comments
  - bugs