COMP 202
Introduction to Computing 1

WELCOME

Focus of the Course
• Introduction to programming using Java
• Aimed at students with LITTLE or NO background in programming and knowledge of Computer Science

Course Outline
• Week 1-2: Introduction
• Weeks 2-3: Java in 2 weeks
  – Variables and assignments
  – if-then-else and loops
  – the class concept
• Week 4: Data and Expression and classes in detail
• Weeks 5-6: Control structures
• Week 7-8: Object-oriented programming
• Week 9: Arrays
• Week 10-13: Advanced Topics

Week 1
• Administrative
• What is programming?
• How does a computer work?
• Our first programs
Instructor Coordinates

- Clark Verbrugge
  - clump@cs.mcgill.ca
- McConnell Eng. Building, Room 230
  - www.sable.mcgill.ca/~clump
- Office Hours:
  - Wednesday 10:00 am - 11:00 am
  - Friday 10:00 am – 11:00am
- Course website:
  - www.cs.mcgill.ca/~cs202

Structure

- Placement Quiz
- 3 hours lecture per week
- 9 tutorials
- Lab TA-Office hours
  - 7 hours per week
  - Times will be announced next week
  - TA's will patrol and be in a nearby room

Tutorials: TBA

- Tutorial 1 is next week
  - Several sessions
  - Date, place TBA
- Tutorial 1:
  - SOCS accounts, basics of Unix
  - WebCT
  - DrJava
- Tutorial 2 and 3:
  - basics of programming
- Tutorials 4, 6, 7, 8:
  - advanced programming
- Tutorials 5 and 9:
  - preparation for midterm, exam

What you have to do

- 5 Assignments
  - Part 1
    - Not graded
    - Simple questions and small exercises to reinforce what was learned in class and help study
    - TAs will help during lab hours to solve the problems
  - Part 2
    - Graded
    - Programming tasks
- Midterm + Final
- Labs: Optional, but highly recommended
Grading Scheme

- Assignments: 30%
  - Assignments 1 through 4 have the same weight (5%)
  - Assignments 5: 10%
  - Late penalty
- Midterm: 20%
  - Date to be set soon
- Final: 50%

Communication

- WebCT Vista
  - Lecture notes
  - Assignments
  - Frequently asked questions
  - Discussions
  - Announcements
  - …
- Notes, assigs also on website:
  - www.cs.mcgill.ca/~cs202
Selection Screen

Course List

The Course

Discussions
Assignments

What is programming?

What do other sciences do?
• Life Science:
  – Example: study how a cell works
  – cell is something complex that already exists
  – researcher tries to figure out how this complex cell works
    • get background knowledge
    • learn laboratory skills
    • make hypothesis
    • run many experiments to see whether hypothesis works
• Engineer
  – Example: build a bridge
  – Create something new
    • Must follow many rules --> must work
    • Must know the math, materials, physics,…
  – What is more complex: a cell or a bridge?

What do Computer Scientists do?
• Example: build the web-portal for an online bookstore
• Create something new
  – Webpages
  – Functionality:
    • browse with keywords
    • Maintain your shopping card
    • Perform checkout
• What is more complex: a cell or a piece of software?
Let’s start small:
Recipe for Scrambled Eggs

- Ingredients: two eggs, tablespoon of oil, salt
- Instructions:
  - Add oil to pan
  - Heat pan on stove
  - Crack eggs into pan
  - Add salt
  - Mix until light and flakey
- Output: scrambled eggs

What if we did not follow the order?

A recipe is a series of steps

Currency Exchange

1. Input:
   - Amount
   - Source Currency
   - Desired Currency
- Instructions
  - Look up in table current exchange rate for the selected currencies
  - Calculate result as amount * exchange rate
- Output: result

What is a program?

- A program solves a particular task
- A program is a list of instructions
- The list must be ordered correctly
- A program has inputs and outputs
- Each instruction tells the computer to do something (an action, a calculation, a comparison)

What is software?

- Software = Program?
- Software typically consists of many subcomponents / modules / subprograms
- Each subprogram solves a particular task
- Software also includes the data to be used and manipulated
  - input / output data
  - internal data (e.g., table with exchange rates)
The bank machine: functionality

- Withdraw money from checking account
- Show balance on checking account
- Show balance on saving account
- Transfer money from checking account to saving account
- Transfer money from saving account to checking account
- Deposit money on checking account
- Deposit money on saving account
- Make bill payment from checking account
- …

The bank machine: Interaction

- Login (sub-program)
- Task menu (each being a sub-program)
  - Withdraw
  - Balance
  - Transfer
  - Deposit
  - Bill payment
- Once task is selected walk-through for specific task on hand
  - requires different input data, produces different output
- Allow many tasks in one session
- logout

The bank machine: login

- Input: pin
- If pin is correct
  - Result: go to task menu
- Else get new pin as input
- If pin is correct
  - Result: go to task menu
- Else get new pin as input
- If pin is correct
  - Result: go to task menu
- Else print warning, do not release card

Bank machine: money transfer

- Input?
- Instructions?
- Output?
Goal of this Course

- Translate a complex task into the software that executes this task
  - *Structure* the problem space into many small sub-tasks
  - Write programs for each subtask
  - *Connect the programs* to build software
- At the end of course, you will be able to write a program that simulates a bank machine (and many other things)
  - Of course, since we don’t have the machines themselves, no “real” money and “real” cards are involved

The Java Programming language

- 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*
  - Java was created by Sun Microsystems, Inc.
  - It was introduced in 1995 and has become quite popular
  - It is an object-oriented language

Our first Java Program

```java
public class MyFirstProgram {
    public static void main (String [] args) {
        // let's say hello
        System.out.println("hello world!");
    }
}
```

Output: hello world

Program Structure (for now)

- Each program is a file
- Each program describes one *class*
- A *class* describes a real-life concept
  - Bank machine, Calculator, System.out, MyFirstProgram
- A *class* has data and methods (functions, operations)
- Example methods
  - *main method* (most classes have one main method)
  - *println*
    - In the example, we provide an implementation for *main* and we use *println* (somebody else has already programmed this method; we simply call it)
- A method contains *statements*
  - Statements are the instruction steps that are executed when the program runs
- Ignore *public, static, void, args* for now
Java Program Structure

```java
public class MyProgram {
  // class header: The name of the class
  class body

  // Important: The class header name MUST be the same name as the file name:
  MyProgram.java
}
```

Comments can be added almost anywhere

```
public class MyProgram {
  
  // method header
  method body

  // Important: The class header name MUST be the same name as the file name:
  MyProgram.java
}
```

Syntax is important

- Natural Languages are ambiguous
- Programming languages may not!
- Examples of Java syntax rules
  - The class name is the same as the file name
  - Parenthesis
- Comments
  - Describe purpose and programming steps
  - Do not affect what the program does
  - Syntax:
    - `//` this is a comment
    - `/*` this is a comment `*/`

DrJava
Compiling and running a program in DrJava

- Type in your program
- Save the program
  - Store all your files in one directory for now
  - Give the program the same name as the class
- Compile the program
  - this produces a .class file
  - Translates the program into something the computer can understand and execute
- Run the program
- Observe the result and adjust the program if necessary

How a computer works

Hardware and Software

- Hardware
  - the physical, tangible parts of a computer
  - keyboard, monitor, wires, chips, mouse
- Software
- A computer requires both hardware and software
- Each is essentially useless without the other

The Personal Computer
CPU (Central Processing Unit)

- Performs instructions
  - Arithmetic operations
  - Much more fine-grained than what we have seen so far
  - Controls the information / data in a computer

Memory

- Memory holds the data
- Main memory (most of it called RAM):
  - Data has to be in main memory so that CPU can access it
  - Volatile: lost when program exits; computer shuts off
- Disk / CD
  - Persistent
  - This is where you keep the data for long-term storage

Interaction
Main Memory

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

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

Each cell contains a data value, e.g. 22

Bits and Bytes

• Bit -- most basic unit of memory
  – 1 or 0, on or off
• 1 Byte = 8 bits
• In a computer, data values are stored as a sequence of bits

Program execution

• A program tells the CPU how to manipulate and/or move information
• Programming is about processing information
  – Take some input, manipulate it in some way, and produce a particular output

Program / CPU / Memory

• Example:
  – Input read a number from keyboard
  – Add 1 to it
  – Output it on screen
In Java

Improving our understanding

- Compile the program
  - The CPU does not understand the Java language!
  - Java Language is translated into language that the CPU understands
- To build this “program” we’ll need 3 classes:
  - Each class fulfills a task
  - System.out: allows us to print things on the monitor
  - Scanner: allows us to read input from keyboard
  - AddOne: that is the class that implements our task!

Variables

- A variable is a placeholder for a cell/location in main memory (or a consecutive block of cells)
- Each variable has a type
  - For now: integers (int)
- Each variable has a name
  - This is how we can refer to the data
  - Much easier than using an address
- Each variable has a value
  - The content of the cell (or set of cells) the variable is referring to
  - An int has 4 Bytes, thus it uses 4 consecutive cells

Variable Declaration

- Variable Declaration
  - Typically at begin of method
  - Must indicate type, name
    - int input;
  - Can indicate several variables in one statement
    - int input, output;
  - Can additionally indicate value
    - int input = 1;
Using a Variable

• Variable Assignment Statement
  – Throughout the program we can assign values to a variable
    • `input = scan.nextInt();`
    • `output = input + 1;`
• The assignment operator is the `=` sign
• The expression on the right is evaluated and the result is stored in the variable on the left
  – The old value of the left variable is overwritten
• Reading
  – We can read the value
    • `output = input + 1;`
    • `System.out.println(output);`

println so far

• `println` takes one input
  – A character string: `println("hello world")`
  – The value of a variable: `println(output);`
    • This value is automatically converted into a character string that can be printed

What we learned so far

• What is a program?
• How does a simple Java program look like?
• How is a simple Java program executed on a computer?
• Fundamentals:
  – A program has operations/statements/instructions
    • They are executed by the CPU
  – A program uses data
    • Data is stored in main memory
    • Data is accessed via the variable concept
  – Different tasks are implemented via different classes