**Computer Science 112 Spring 2013**

Lab 2

**Due by in class one week from today.**

**Introduction**

In lab 1, we went over printing out characters to the screen using System.out. Today you will be introduced to the use of algorithms, different types of errors, input statements, JOptionPanes, if statements, and character comparisons.

For part 3, you will work backwards and use pseudocode to write an algorithm from the Java code. This will give you practice with algorithms while allowing you to explore and understand a little Java code before we begin learning too much of the Java programming language. You will also need to test out this program to ensure the correctness of the algorithm and code. You will need to develop test data that will represent all possible kinds of data that the user may enter.

You will also be debugging a program. There are several types of errors. In this lab, you will encounter syntax and logic errors. We will explore runtime errors in lab 1 part 2.

Syntax Errors—errors in the “grammar” of the programming language. These are caught by the compiler and listed out with line number and error found. You will learn how to understand what they tell you with experience. All syntax errors must be corrected before the program will run. If the program runs, this does not mean that it is correct, only that there are no syntax errors. Examples of syntax errors are spelling mistakes in variable names, missing semicolon, unpaired curly braces, etc.

Logical Errors—errors in the logic of the algorithm. These errors emphasize the need for a correct algorithm. If the statements are out of order, if there are errors in a formula, or if there are missing steps, the program can still run and give you output, but it may be the wrong output. Since there is no list of errors for logic errors, you may not realize you have errors unless you check your output. It is very important to know what output you expect. You should test your programs with different inputs, and know what output to expect in each case. For example, if your program calculates your pay, you should check three different cases: less than 40 hours, 40 hours, and more than 40 hours. Calculate each case by hand before running your program so that you know what to expect. You may get a correct answer for one case, but not for another case. This will help you figure out where your logic errors are.

Run time errors—errors that do not occur until the program is run, and then may only occur with some data. These errors emphasize the need for completely testing your program.

Put the answers to the questions 1, and 2 in separate block comments at the top of the java file for part 3. Part 4 should be answered in its own java file. Both java files should be zipped together and submitted on moodle under “assignment 2.”

1. **True or False**
   1. The following line of code contains a syntax error:

System.out.println(This is a test);

* 1. The following line of code contains a syntax error:

int x = 7;

* 1. The following is a run-time error:

System.out.println(“Twice the given integer is ” + x\*2);

* 1. Applications can be run from a website. Applets run on the computer.
  2. Programs can be written, compiled, and run via DrJava without ever needing to use the Terminal or typing lines like javac MyFirstProgram.java
  3. The name of the class must match the filename, e.g. MyFirstProgram.java.
  4. The following are all primitive data types: int, double, char, String, boolean
  5. Print statements are a great way to test for logical errors
  6. One can use == to compare two characters
  7. Every block of if-else-if statements must end with an “else” to catch all remaining cases

1. **Short Answer**

Determine if there is an error in each of these situations. If so, what type is it (syntax, logical, or run-time)? If there is an error, replace the wrong line with a correct one. If the program would print something to the screen, write what will be printed. Feel free to run the code in DrJava if you get stumped, but try to figure it out without DrJava first.

// This declares an integer x, sets x to be 7, and prints

// the value of x

int x;

x == 7;

System.out.println(“x”);

// The goal of this code is to print “e3” to the screen

char c = 'e';

char d = '3';

String s = c+d;

System.out.println(s);

/\* This takes an integer x, computes x+2, and prints out the equation. So if

x were 7, this would print “x + 2 = 7”

\*/

int y = x+2;

System.out.println(x+ “ 2 = ” + y);

// This code should test if 1 is equal to 2

int x = 1;

int y = 2;

if (X == y){

system.out.println(“yes, 1 equals 2”);

else if (x != y) {

System.out.println(“no, 1 does not equal 2”)}



// dayOfWeek is an integer. 0 is Monday, 1 is Tuesday, …, 6 is Sunday

if (dayOfWeek == 4)

System.out.println(“The weekend begins after work!”);

else if (dayOfWeek > 5)

System.out.println(“Hooray, we can sleep in!”);

else

System.out.println(“It's not the weekend, so we have to do work”);

1. **Short Program**

Create a program which uses the following code. Follow the comments to complete the program.

// Import the scanner class here

import …......;

public class compareCharacters

{

public static void main(String[] args)

{

// Follow the example on page 88:

String input;

char c;

char d;

// Declare a scanner

…........................;

// Ask for a character

System.out.print(“Enter a character: ”);

// Set input to be the scanner line read in, set c = first char in input

// Repeat for a second character d

// if c and d are equal, print “They are equal”

// Otherwise, print “They are not equal”

} // end of main

} // end of class

Test your code on several examples, listing them in the comment section of your program.

1. **Long program**

Write a program which uses JOptionPane input dialog to ask the person running the program for a letter grade (A,B,C,D,E,F – no plusses or minuses). Use an if/else if/else statement to print out a message using a JOptionPane message dialog for each of them. Feel free to be as funny as possible. Since input dialogs take in a string of characters, you will need to look up how to use the “charAt” method to get the first character. Further, you should ignore the case of the letter. To do this, you should use the logical operators we talked about in class to cover both possibilities. In addition, use the “trailing else” to send a message to the user that they inputted an invalid character.