Homework 5

This assignment covers methods and classes. This homework also assumes understanding of the concepts that are covered previously. The one programming exercise is worth 70 points. If you need help, ask the TAs, ask Ed, email the teaching assistants or Ed, come to office hours, etc. The rubric for this assignment is available on the course website.

Questions – 30 points. This part adds up to 31, so 1 point extra credit.

For the questions listed below, upload the single “Homework 5 Answer Sheet.docx” document to your Canvas account that includes your answers.

Multiple Choice and True/False, questions, 1 point each

1. (1 point) True or False: You terminate a method header with a semicolon.

Write answer on answer sheet.

2. (1 point) True or False: A parameter variable’s scope is the entire program that contains the method in which the parameter is declared.

3. (1 point) True or False: When code in a method changes the value of a parameter, it also changes the value of the argument that was passed into the parameter.

4. (1 point) True or False: The contents of a String object cannot be changed.

5. (1 point) True or False: No two methods in the same program can have a local variable with the same

6. (1 point) A class is analogous to a(n) \_\_\_\_\_\_\_.

a. house

b. blueprint

c. drafting table

d. architect

7. (1 point) This is a class member that holds data.

a. method

b. instance

c. field

d. constructor

8. (1 point) This is a method that is automatically called when an instance of a class is created.

a. accessor

b. constructor

c. void

d. mutator

9. (1 point) Two or more methods in a class may have the same name, as long as this is different.

a. their return values

b. their access specifier

c. their parameter lists

d. their memory address

Short Answer 1 point each

10. (1 point) What is the difference between a class and an instance of a class?

11. (1 point) Is it a good idea to make fields private? Why or why not?

12. (1 point) What is the purpose of the new key word?

13. (1 point) What do you call a constructor that accepts no arguments?

Find the Error, Rewrite correctly, 2 points

**14**. (2 points) Look at the following method header:

public static void showValue(int x)

The following code has a call to the showValue method. Find the error.

int x = 8;

showValue(int x);

Find the Error, Rewrite correctly, 2 points

**15**. (2 points) Find the error in the following class:

public class MyClass {

private int x;

private double y;

public void MyClass(int a, double b) {

x = a;

y = b;

}

}

**16**. (2 points) Assume that the following method is a member of a class. Find the error. Rewrite correctly, 2 points

public void total(int value1, value2, value3) {

return valuel + value2 + value3;

}

**17**. (2 points) Find the error in the following class. Rewrite correctly, 2 points:

public class TwoValues {

private int x, y;

public TwoValues() {

x = 0;

}

public TwoValues() {

x = 0;

y = 0;

}

}

Algorithmic Workbench 2 points each

**18**. (2 points) What will the following program display?

public class ChangeParam {

public static void main(String[] args) {

int x = 1;

double y = 3.4;

System.out.println(x + " " + y );

changeUs (x, y);

System.out.println(x + " " + y );

}

public static void changeUs(int a, double b) {

a = 0;

b = 0.0;

System.out.println(a + " " + b);

}

}

**19**. (2 points) A program contains the following method:

public static void display(int arg1, double arg2, char arg3) {

System.out.println(“The values are “ + arg1 + ", " + arg2 + ", and " + arg3);

}

Write a statement that calls this method and passes the following variables as arguments:

char initial = 'T';

int age = 25;

double income = 50000.00;

**20**. (2 points) Look at the following partial class definition, and then respond to the questions that follow it:

public class Book {

private String title;

private String author;

private String publisher;

private int copiesSold;

}

a. Write a constructor for this class. The constructor should accept an argument for each of the fields.

b. Write accessor and mutator methods for each field.

c. Draw a UML diagram for the class, including the methods you have written.

**Programming Question – Casino and Gambler.java - 70 points**

This programming section of this assignment requires you to submit TWO .java files. Please read all of the instructions, before you begin writing your code. Details were also provided during lecture.

For this programming question, you'll be given a skeleton of a program, Casino.java, that contains pseudocode ONLY. The java file is well documented. Follow the instructions in the java file (from top to bottom), to write the code needed to finish this programming task. You'll also need to write a second file, called Gambler.java. It is a class representing a person at a casino. The Gambler.java class has 4 fields, 6 methods (two of which take an argument, and four of which do not), and a non-default constructor, that takes a single argument.

The setup is the following: The Casino.java program instantiates four objects of the class Gambler, and for each of them declares a reference variable. The Gambler class has fields for a person's name, his/her winning percentage (how “lucky” the gambler is, ranging from 0 (always loses) to 1 (always wins)). In the Casino.java program, a while loop, with true as the condition, repeatedly prints out each gambler's name, how much money he/she has spent, how much money he/she currently has, and the gambler's net gain (or loss). At each iteration of the while loop, the program asks, “How much money should each gambler bet? Enter 0 to quit.” If the user enters, via keyboard, 0, then a break statement is invoked, and the while loop (and hence program) terminates. If a non-zero value is entered, then the “gambleAnotherRound” method for each of the four Gambler objects is invoked, which either increments or decrements each gambler's winnings, based on whether a randomly generated decimal number is more or less than a gambler's winning percentage. See the skeleton of Casino.java, which contains pseudocode of the entire program.

The Gambler.java class has four fields, gamblerName, winningPercentage, dollarsSpent, and dollarsWon. The class has a method, setWinningPercentage, that takes as input the single argument winningPercent of type double. The class has a second method, gambleAnotherRound, that takes as input the single argument dollarsBet, of type double. The method gambleAnotherRound :

• Generates a random number

• checks if that number is more or less than the Gambler's winningPercentage; depending on the outcome, the method prints out to the console “won” or “lost”. If a gambler wins a round, the Gambler's dollarsWon is incremented by 1.5 times the Gambler's bet.

Five additional methods, getGamblerName, getDollarSpent, getWinningPercentage, getDollarsWon, and getNetProfits, take no input parameters, and return a String, double, double, double, and double, respectively. The Gambler class non-default constructor takes a single argument, name, of type String. See the skeleton of Gambler.java, which explains each field, and the methods, in general. The UML diagram for the Gambler class is shown below:

|  |
| --- |
| Gambler UML, Unified Modeling Language |
| - gamblerName : String  - winningPercentage : double  - dollarsSpent : double  - dollarsWon : double |
| + getWinningPercentage() : double  + setWinningPercentage(winningPercentage : double) : void  + gambleAnotherRound(dollarsBet : double) : void  + getGamblerName() : String  + getDollarsSpent() : double  + getDollarsWon() : double  + getNetProfit() : double  + Gambler(name : String) : |

A sample invocation of the program is shown in Figure 1, in which there are three rounds of gambling. In the first round, $30 is bet by each player. In the second and third rounds, $16 and $100 is bet by each player. Each player has a different chance of winning, and by the end of the three rounds of betting, three of the gamblers have a negative net gain, and the player Sneezy, has a net of $108.

|  |
| --- |
| Here are the gamblers, and their winning percentages:  Grumpy, winning percentage :0.61  Sleepy, winning percentage :0.43  Sneezy, winning percentage :0.51  Cheaty, winning percentage :0.31  Gambler Grumpy; spent $0.0; current $100.0; net $100.0  Gambler Sleepy; spent $0.0; current $100.0; net $100.0  Gambler Sneezy; spent $0.0; current $100.0; net $100.0  Gambler Cheaty; spent $0.0; current $100.0; net $100.0  How much money should each gambler bet? Enter 0 to quit 30  Grumpy lost  Sleepy won  Sneezy lost  Cheaty lost  Gambler Grumpy; spent $30.0; current $70.0; net $40.0  Gambler Sleepy; spent $30.0; current $145.0; net $115.0  Gambler Sneezy; spent $30.0; current $70.0; net $40.0  Gambler Cheaty; spent $30.0; current $70.0; net $40.0  How much money should each gambler bet? Enter 0 to quit 16  Grumpy lost  Sleepy won  Sneezy won  Cheaty lost  Gambler Grumpy; spent $46.0; current $54.0; net $8.0  Gambler Sleepy; spent $46.0; current $169.0; net $123.0  Gambler Sneezy; spent $46.0; current $94.0; net $48.0  Gambler Cheaty; spent $46.0; current $54.0; net $8.0  How much money should each gambler bet? Enter 0 to quit 120  Grumpy lost  Sleepy lost  Sneezy won  Cheaty lost  Gambler Grumpy; spent $166.0; current $-66.0; net $-232.0  Gambler Sleepy; spent $166.0; current $49.0; net $-117.0  Gambler Sneezy; spent $166.0; current $274.0; net $108.0  Gambler Cheaty; spent $166.0; current $-66.0; net $-232.0  How much money should each gambler bet? Enter 0 to quit 0 |
| Figure 1: sample output of the program Casino.java |

Submit your four files ( separately or as zip):

“Homework 5 Answer sheet.docx”

“Casino.java”

“Casino output.docx”

“Gambler.java”