

AP Comp Sci A/B Mr. Hanley

Assignment 0₂/0₈/0₁₆ Summer Assignment

Binary

--	--	--	--	--	--	--	--	--	--

Ones Comp

--	--	--	--	--	--	--	--	--	--

Twos Comp

--	--	--	--	--	--	--	--	--	--



"You can't just punch in 'let there be light' without writing the code underlying the user interface functions."

Welcome to the AP Computer Science Program!

This is a challenging and fun course as we delve deeper into computer science.

Attached you will find summer assignments to be completed by the first day of school next year.

The two articles are from the Sun page and they describe the basic concepts of classes and objects. Here are the URL's

<http://java.sun.com/docs/books/tutorial/java/concepts/object.html>

<http://java.sun.com/docs/books/tutorial/java/concepts/class.html>

String* Arrays* ArrayList* Sets* Maps* Artificial Intelligence* Heaps* Files* Video Games* Short circuit evaluation

*Dynamic Memory *Big O Notation *Stacks* Linked Lists* Binary Trees* Selection Sort* Insertion Sort* Hashing* Priority Queue* Collisions*

1. Prepare a 6-8 slide powerpoint presentation on the articles describing the differences. (If you don't have powerpoint, download open office and use that presentation tool) If necessary, use the internet and computer books to explore the differences between classes and objects, etc.

The following programs can be created using either the console, SenezConsole or Swing.

2. Write a program that takes as inputs the lengths of three sides of a triangle and displays; first, whether or not the 3 values represent a triangle and then whether the triangle is scalene, isosceles, and equilateral. Useful facts:
 - In a triangle, the longest side must be less than the sum of the other two sides.
 - A scalene triangle has all sides unequal.
 - An isosceles triangle has two sides equal. . An equilateral triangle has all sides equal.

HINTS for Testing: When testing, make sure you test with all sides the same and your program doesn't report the triangle to be both isosceles and equilateral.

3. In the game of craps, a player provides an initial bankroll and bets from this amount on each roll the dice. On each roll, the sum of the faces is taken. The outcomes are as follows:
 - If 7 or 11 is rolled, the player wins.
 - If 2, 3, or 12 is rolled, the player loses.
 - Otherwise, the number rolled becomes the player's point. The player rolls the dice repeatedly until the player wins by making point (getting the same number as on the first roll) or loses by crapping out (getting a 7).
Design and implement a craps machine that allows the user to play craps. This machine should be defined as a new class. The interface accepts an amount of money representing an initial bankroll. Before each roll of the dice, the user must make a bet. The interface should control the user's options by enabling and disabling the appropriate command buttons. At the end of the game, program should display the amount of the user's current bankroll (after adding the gains and deducting the losses).
- Examples of dice roll sequences and results
 - i. Roll a 7 on first roll -> WIN
 - ii. Roll an 11 on first roll -> WIN
 - iii. Roll a 2 on first roll ->LOSE
 - iv. Roll a 3 on first roll -> LOSE
 - v. Roll a 12 on first roll -> LOSE

- vi. Roll a 5, 9, 12, 6 and then a 5 to win since you rerolled the point
- vii. Roll a 5, 9, 12, 6 and then a 7 to lose since you rolled a 7 when attempting to reroll the point

Roll 8, 2, 3, 11, 12, 9, 3, 3, 5, 9, 10, 8 to win since you rerolled the point

HINTS for Testing: If you seed the Random number generator object, then you will get the same sequence of pseudo random numbers each run of the program. For example, `Random r = new Random(100);` will cause all calls to `r.nextInt(n);` to give the same sequence. This helps you find the repeated case you are looking for without testing forever. Be sure you've covered all cases.

NOTE: Console programmers will be writing a loop while Swing users will have to keep track of whether it is a reroll or initial roll. Swing utilizes its own loop to keep checking for any events that might occur and report those events to the program handlers. Therefore, although there is a loop, it has been created by the authors of the Swing GUI developers. To deal with this, use a global variable inside your Frame to determine if the next roll is an initial roll or reroll.

For example,

```
public CrapsFrame extends JFrame implements ActionListener {  
    boolean firstRoll = true;  
  
}
```

This variable can be used to inform the `actionPerformed` method inside the `CrapsFrame` when a roll is a fist roll and when it is a reroll. Just flip the boolean variable.

4. A perfect number is a positive integer such that the sum of the divisors equals the number. Thus, $28 = 1 + 2 + 4 + 7 + 14$ is a perfect number. If the sum of the divisors is less than the number, it is deficient. If the sum exceeds the number, it is abundant. Write a program that takes a positive integer as input and displays a message box that indicates whether the number entered is perfect, deficient, or abundant. Your program should define the following two methods:

```
boolean isDivisor (int num, int divisor)  
int divisorSum (int number)
```

The method `isDivisor` returns true if the divisor parameter is a divisor of the number parameter, and false otherwise. The `divisorSum` method uses `isDivisor` to

accumulate and return the sum of the proper divisors of the number parameter. Be sure to design and test the program incrementally, that is, verify that isDivisor works correctly before using it in divisorSum.

From a console program, the three methods might look like this;

```
/* prompts the user for a number and then reports whether the number is
perfect, deficient or abundant , calls the divisorSum method to find the number of
proper divisors for any number*/
```

```
public static void main(String[] args) {

}
```

```
/* takes in an integer and calls isDivisor in a loop to determine whether or not
each possible divisor evenly divides the number*/
```

```
public static int divisorSum (int number){

}
```

```
/* takes in an two integers, returns true if the second is a divisor of the first*/
```

```
public static boolean isDivisor (int num, int divisor) {

}
```

5. A standard experiment is to drop a ball to see how high it bounces. Once the "bounciness" of the ball is determined, the ratio gives a bounciness index. For example, if a ball dropped from a height of 10 feet bounces 6 feet high, the index is 0.6 and the total distance traveled by the ball is 16 feet after one bounce. If the ball continues bouncing, the distance after two bounces would be $10 + 6 + 6 + 3.6 = 25.6$ feet. Note that the distance traveled for each bounce is the distance to the floor plus 0.6 of that distance as the ball comes back up. Write a program that takes as inputs the initial height of the ball (in feet), the index of the ball's bounciness, and the number of times the ball is allowed to continue bouncing. The program should output the total distance traveled by the ball. At some point in the process, the distance traveled by the ball after a bounce might become negligible, for example, less than 0.000001 feet . If that stage is reached,

terminate the process and output the total distance. (This could happen if for example, we attempted to drop a ball for 10,000 bounces).

6. Create an account on the javabat.com web site. This site allows you to type in the bodies of methods and see if they are correct online. Share your account out to mrhanleyc@hotmail.com under the prefs for javabat. **Your first and last name in the preferences must be your actual first and last name so I don't have to guess who is who.** Complete the following javabat problems;
 - Complete the following **Warmup 1** problems and save them on javabat so I can verify, comments helpful but not required
 - i. sleepIn, monkeyTrouble, sumDouble, diff21, parrotTrouble, makes10, icyHot, in1020, hasten, loneTeen, intMax, close10, in3050, max1020, lastDigit
 - Complete the **Logic 1** problems
 - Complete the **Logic 2** problems

Project Name	Summer Assignment
Class 1 Name	TriangleCalculator
Class 2 Name	PerfectNumberFinder
Class 3 Name	Craps
Class 4 Name	BounceCalculator

Rubric	
TriangleCalculator	25
PerfectNumberFinder	25
Craps	25
Bounciness Simulation	25
Powerpoint on Object oriented programming	30
Comments	10
JavaBat Problems	100
TOTAL	240