Name _____
Date _____

AP Computer Science A
Coffee Pot

Directions: Create a CoffeePot class according to the specifications below. Then, create a tester class to test the functionality of your CoffeePot.

CoffeePot should have two instance variables:

- cups: an integer to keep track of the number of cups of coffee left in the pot
- blend: a String to hold the name of the coffee blend

CoffeePot should have constructors:

- the default constructor should build a full pot of coffee (12 cups) with "Dunkin Donuts Original Blend"
- include 2 other constructors that are appropriate for this program

CoffeePot should have these additional methods:

- pourCup: pours a single cup of coffee
- fillPot: fills the pot with a specific blend (should have a String parameter)
- fillPot: fills the pot with the current blend
- getCups: gets (returns) the number of cups currently left in the pot
- getBlend: gets (returns) the name of the current blend of coffee

Be creative! What other functionalities can you add to CoffeePot? Add them to your program.

Use a tester method to test the CoffeePot class. You are welcome to incorporate user input, or you can simply create a CoffeePot object and call methods accordingly.

Name:
Date:

AP Computer Science A
Car Lab

Directions: Create a class called Car with the specifications listed.

1. With the Car class, you should have instance variables (fields) that represent year manufactured, miles driven, manufacturer name, and model name.

2. You should have a constructor that has parameters for all of the fields.

3. You should provide the following setter/mutator methods for updated the data values:
   a. setYearManufactured with a parameter for year
   b. setMilesDriven with a parameter for miles
   c. setManufacturerName with a parameter for name
   d. setModelName with a parameter for name

4. You should provide the following getter/accessor methods for obtaining these data values:
   a. getYearManufactured
   b. getMilesDriven
   c. getManufacturerName
   d. getModelName

5. It turns out that a car might be sold to a recent immigrant from Canada. Therefore, you want to be able to report the distance driven in kilometers instead of miles. Add a method called getKilometersDriven that returns the distance driven in kilometers instead of miles. (Useful information: 1 mile = 1.6094 kilometers)

6. Create another class called CarTester, which contains your main method. Test each of the Car methods.

Rubric

	3	**2**	**1**	**0**
Instance Variables	All instance variables are correctly declared.	2-3 instance variables are correctly declared.	1 instance variable is correctly declared.	Instance variables are not declared.
Constructor	Constructor has the correct parameters and properly initializes the instance variables.	Constructor has the correct parameters but there are errors in initializing the instance variables.	Constructor has parameters and initializes instance variables but with errors or omissions throughout.	Constructor does not exist in the class.
Setter/ Mutator Methods	All setter methods are correctly implemented.	2-3 setter methods are correctly implemented.	1 setter method is correctly implemented.	Setter methods are not correctly implemented.
Getter/ Accessor Methods	All getter methods are correctly implemented.	2-3 getter methods are correctly implemented.	1 getter method is correctly implemented.	Getter methods are not correctly implemented.
Kilometer Conversion	getKilometersDriven properly converts the distance and has an appropriate return value.	getKilometersDriven converts the distance incorrectly but has a return value.	getKilometersDriven converts the distance incorrectly and/or does not have a return value.	getKilometersDriven does not exist in the class.
CarTester	Tester class properly tests all methods of the Car class.	Tester class tests the constructor and 5-8 methods of the Car class.	Tester class tests the constructor and 1-4 methods of the Car class.	Tester class does not test the constructor or the methods of the Car class.
Style	Classes follow Java style guidelines and include appropriate variable names, comments, etc.	Classes follow Java style guidelines and include 1-2 questionable variable names, comments, etc.	Classes follow Java style guidelines and include multiple questionable variable names, comments, etc.	Classes are missing comments and/or have inappropriate variable names.

Name:
Date:

APCS Taco Lab (Extra)

Directions: You are writing a program for a taco.

1. Instance Variables:
   - filling
   - toppings
   - shell
   - price
   - day of week

2. Constructor:
   - create a default constructor
   - create a constructor with parameters for all of the instance variables except the price, which should have a default start price

3. Methods:
   - setFilling(): String parameter
   - addTopping(): String parameter, use concatenation (+) to add on to the list of toppings. This method should also add $0.50 to the price of the taco
   - setShell(): String parameter
   - TacoTuesday (): boolean parameter for day of the week; if it is Tuesday, change the price to reflect a 10% discount
   - getter methods for filling, toppings, shell, and price

4. Tester Method
   - create a tester method to make sure that all of the constructors and methods you created work properly

Name _____
Date _____

AP Computer Science
Carnival Game Lab

Create a class CarnivalGame that has two integer instance variables that represent dice and store the value of each die.

The constructor should not take any parameters and should assign values to the two dice by using the Math.random() method.

The class should include the following methods:

- rollDieOne – "rolls" and assigns a new value to the first die by using Math.random()
- rollDieTwo – "rolls" and assigns a new value to the second die by using Math.random()
- roll – "rolls" and assigns new values to both die by using Math.random()
- getDieOne – returns the value of the first die
- getDieTwo – returns the value of the second die
- winnings – calculates and returns the player's winnings according to the following rules:
    - If the two dice have the same value, the number of cents that the player receives is 25 times the product of the two dice
    - If the two dice do not have the same value, the number of cents that the player receives is 10 times the positive difference of the two dice
    - The winnings should be returned as a formatted string (e.g. $12.41). Use integer division and % to determine the number of dollars and cents that the player wins.

Create a tester class with a main method that creates a CarnivalGame object. After you create the CarnivalGame object, get user input to determine whether the user wants to keep the original values of the die, roll the first die, roll the second die, or roll both dice. (You can use a menu and an integer input to do this.) Call the appropriate method. After the appropriate method has been called, you should display the player's winnings in a formatted statement. The statement should read:

You rolled a <insert die value> and a <insert die value>. Your winnings are <insert winning>.

Rubric

	3	**2**	**1**	**0**
Instance Variables	Instance variables are properly	Instance variables are declared but with 1 error	Instance variables are declared but with 2 or more errors	Instance variables are not declared
Constructor	Constructor is correctly declared and properly assigns values to both dice	Constructor is correctly declared but there is error in using Math.random() to assign values to the instance variables	Constructor is declared but with errors and there is error in using Math.random() to assign values to the instance variables	Constructor is not declared or instance variables are not initiated
rollDieOne/ rollDieTwo	Methods are declared properly and correctly reassigns the value of each die respectively using Math.random()	Methods are declared properly but there is error in using Math.random() to reassign the value of each die respectively	Methods are declared but with error and there is error in using Math.random() to reassign the value of each die respectively	Methods are not declared or values of the dice are not reassigned
Roll	Method is declared properly and correctly reassigns the values of both dice using Math.random()	Method is declared properly but there is error in using Math.random() to reassign the values of both dice	Method is declared but with error and there is error in using Math.random() to reassign the values of both dice	Method is not declared or the values of both dice are not reassigned
getDieOne / getDieTwo	Methods are correctly declared and return the values of each die respectively	Methods are correctly declared but values are returned with error	Methods are declared but with error and there is error in returning the values of the instance variables	Method is not declared or the values are not returned
Winnings-calculations	Method correctly calculates value of winnings according to the conditions described	Method calculates value of winnings according to the conditions but with 1-2 errors.	Method calculates value of winnings but with multiple errors	Method is not declared or the value of winnings is not calculated
Winnings – return	Method returns a properly formatted String using integer operations	Method returns a properly formatted String but does not use integer operations	Method returns a value but not as a properly formatted String	Method is not declared or does not return a value other than a String
Tester Class	Tester class has a main method with appropriate while loop that allows user choice and prints appropriate outcome	Tester class has a main method with a while loop that has 1 error but allows user choice and prints appropriate outcome	Tester class has a main method but a while loop but has multiple errors in asking for user choice and printing outcome	Tester class does not exist
Style	Classes follow Java style guidelines and include appropriate variable names, comments, etc.	Classes follow Java style guidelines and include 1-2 questionable variable names, comments, etc.	Classes follow Java style guidelines and include multiple questionable variable names, comments, etc.	Classes are missing comments and/or have inappropriate variable names

Name:
Date:

AP Computer Science A
String DNA Lab

Directions: DNA is the foundation of life, and Bioinformatics is the science of analyzing biological data, such as DNA and genetic sequences.

The four bases found in DNA are molecules called drenine (A), cytosine (C), guanine (G), and thymine (T). It has been found that certain conditions are caused by particular genetic mutations that can be found in the DNA sequences.

Your task is to write code that tests for each syndrome for each of the three patients whose DNA sequences are provided in the attached file. You should use String methods to test for each of the syndromes and output the results for each patient.

Howlett's Syndrome is characterized by excessive hair growth, especially on the face, and internal pain in phalangeal region of the hands. Howlett's is represented in DNA by the sequence: **ACGTTCGAGATCGA**.

Parker's Syndrome is characterized by heightened perception, strong visual acuity, and gelatinous discharge from pores in the hands and wrists. Parker's is represented in DNA by the sequence **GTACCAGTACGATCAG.**

Banner's Syndrome is characterized by increased irritability and unpredictable behavior when under stress. Banner's is represented in DNA by the sequence **CGTAGATCATGACGA**.

Danvers' Syndrome is characterized by repeated bouts of memory loss and authoritative personality traits. Danvers' is represented in DNA by the sequence **ACTGGTCA** only if it found in the second half of one's DNA. If any part of the sequence is in the first half of one's DNA, it has no effect.

Stark's Syndrome is characterized by high levels of both intellect and narcissism, along with irregular heartbeat patterns. Stark's is represented in DNA by the sequence **GATCGATGTGCAGACTAGCGAT** only if it is found, in its entirety, in the first half of one's DNA sequence**.** If any part of the sequence is located in the second half of one's DNA, it has no effect.

Romanova's Syndrome is characterized by highly levels of empathy for others and tendencies to exhibit reckless behavior when under duress. Romanova's is a little bit trickier to identify. It is represented by two distinct occurrences of the sequence **GTCAGGAC** in one's DNA. If it occurs only once, it has no effect.

LaForge Syndrome causes congential blindness. You need to look for the following in the first 40 places of the DNA sequence only: **GATCGATGTGCAGACTAGCGAT**. If it is after the first 40 places, it has no effect.

```
String patient1 =
"ACGATCGATCGATCGATGTGCAGACTAGCGATGAGCTAGCTGAGATCGGATGCTAGCTAGCTAGCATCGATCG
ATCGACTACGCTAGCTAGCCTAGCATCGACTGCATCGACTAGCATCGACTAGCTAAGCATCAGCTCGACTAGCA
CGTTCGAGATCGAATCGATGCATCGCATCGACTCGATCGACTAGCGCATCAGTAGCGAT";
String patient2 =
"GATCGACTGGTCAGATGCTAGCTAGCTAGCATCGATCGATCGACTACGCTAGCTAGCCTAGCATCGACTGCAT
CGACTAGCATCGACTAGCTAAGCATCAGCTCGACTAGCATCGATGCATCGCATCCGTAGATCATGACGAGACTC
GATCGACTAGCGCATCAGTAGCGATTCGATGCAACTGGTCATCGCATC";
String patient3 =
"CACTGGTCATAGCTGAGATCGGATGCTAGCTAGCTAGCATCGGTACCAGTACGATCAGATCGATCGACTACGC
TAGCTAGCCTAGCATCGACTGCATCGACTAGCATCGACTAGCTAAGCATCAGCTCGACTAGCATCGATGCATCA
CGTTCGAGATCGAGCATCGACTCGATCGACTAGCGCATCAGTAGCGATGATCGACTAGCGCATC";
```