| | |
|---|---|
| **Started on** | Monday, 8 February 2021, 12:15 PM |
| **State** | Finished |
| **Completed on** | Monday, 8 February 2021, 2:15 PM |
| **Time taken** | 1 hour 59 mins |
| **Grade** | **7.60** out of 15.00 (**51**%) |

Question **1**

Partially correct

Mark 0.50 out of 2.00

Rewrite the following five assignment statements into *a single statement using prefix and postfix increment* and decrement operators as necessary. Assume all variables are int variables.

```
x = y + 1;
y = y - 1;
z = z + 1;
x = x - z;
x = x + 1;
```

Your single statement should begin with:

```
x = ...  ;
```

Answer:

```
x = (++y) - (--y) - (++z) + (++x);
```

✖

The correct answer is: x = (y-- + 1) - ( ++z ) + 1 ;

Comment:
more -1.5

Question **2**

Complete

Mark 2.60 out of 3.00

Convert the following Java code fragment:

```
if (x == 1 || x == 2)
{
        switch (ch)
        {
                case 'a': System.out.println ("Good");
                case 'b': System.out.println ("Luck");
        }
}
else if (x == 3 || x == 4)
{
        if (ch == 'c' || ch == 'd')
                System.out.println ("To");
        else
                System.out.println ("Your");
}
else
{
        switch (ch)
        {
                case 'e':
                case 'f': System.out.println ("Term");
                default : System.out.println ("Test");
        }
}
```

(a) Change all the **if-else** statements to **switch** statements
(b) Change all the **switch** statements to **if-else** statements

Objective after the change remains unchanged. You can assume that the two variables **x** and **ch** used in the code

fragment are properly defined and initialized.

Save your conversion in a Java file called **YourName_Q2.java** and upload this Java file. Alternatively you can copy and paste your answer in the answer box make the statements are properly indented and well aligned.

</> MinZhanFoo_Q2.java

Comment:

Marking scheme of Question 2

conversion of if-else (1.5 marks):

conversion of switch statements (1.5 marks):

More refinement -0.4

Question **3**

Complete

Mark 0.50 out of 2.00

The Student class below will allow student information to be constructed in various ways but is repetitive. How could you use the **this** reference to reduce the amount of repetitious code and make it less error prone?

```
1  class Student {
2      private int id;
3      private double mark;
4
5      public Student (int anID) {
6          id = anID;
7          mark = 100.0;
8      }
9
10     public Student (double aMark){
11         id = 888;
12         mark = aMark;
13     }
14
15     public Student (int anID, double aMark){
16         id = anID;
17         mark = aMark;
18     }
19
20     public Student () {
21         id = 888;
22         mark = 100.0;
23     }
24 }
```

Indicate the statement number(s) to be replaced; and its (their) replacement.

line 5: public Student (int id) {
line 6:  this.id = id;

line 10: public Student (double mark) {
line 12 this.mark = mark;

line 12 this.mark = mark;

line 15: public Student (int id, double mark) {
line 16: this.id = id;
line 17: this.mark = mark;

Comment:

**Marking scheme of Question 3**

**// All three (2 marks)**

**this(100.0, mark);**

**this(id, 888);**

**this(888, 100.0);**

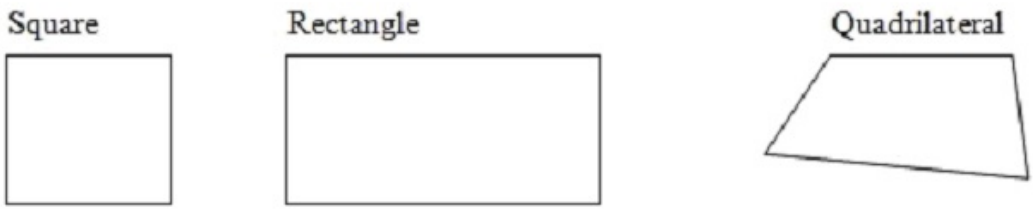
**-1.5**

Question **4**

Complete

Mark 4.00 out of 4.00

A shape with four sides can have fours sides of equal length, as in a square (**SQR**); two pairs of two sides with equal lengths, where each pair has a different length, as in a rectangle (**REC**); or be something with four sides of different length (a normal quadrilateral, **QUA**). You can assume that if the four sides are positive, a shape of four sides can be formed. Examples of theses shapes are illustrated below:




Design a Java program according to the following UML diagram **(No additional instance variables and methods are allowed, i.e. assessor and mutator methods are not required unless you have nothing else to do ...):**


We define four **String** constants **SQR**, **REC**, **QUA** and **NIL** that takes the values ***"Square"***, ***"Rectangle"***, ***"Quadrilateral"***, and ***"No shape"*** respectively. You should fully explore the use of these four constants in your design. Let us talk about the design:

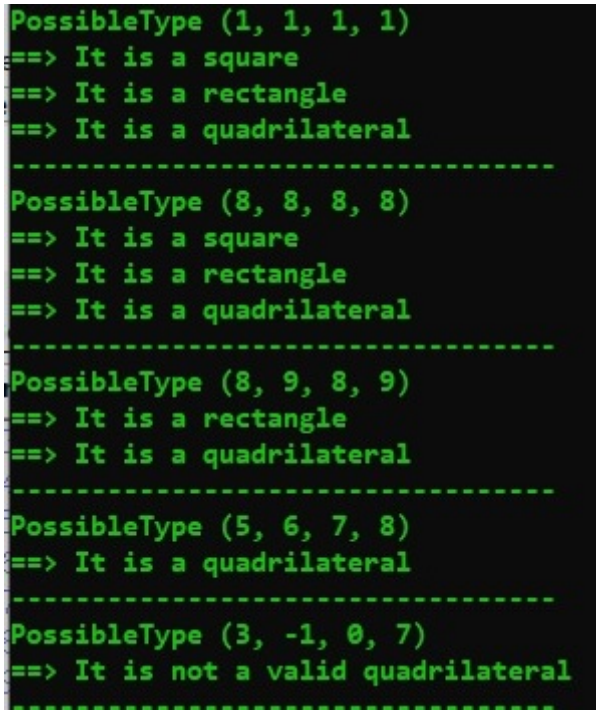
- For the default constructor, you initialize the shape to be a square of all sides equal to 1;

- For other constructor, you need to invoke the **getType** method to initialize the type of 4 sides (the return type is a String object the takes one of the four String constants).

- The **getType** method must be a private method to determine the type (one of the 4 String constants); and

- The **displayInfo** method should display the information of the sides according to the specification listed below. You must use the switch statement in the display.

- In the main method, construct 5 **FourSides** objects (hard code the 4 sides and use the default constructor to construct the 1st object), and display them. Here is the sample output:




The name of the Java program must be **YourName_Q4.java**; and upload this Java file.

**Note that if you don't have time to upload the Java program, you can copy and paste your program in the answer box (auto save).**

</> MinZhanFoo_Q4.java

Comment:

Marking scheme of Question 4

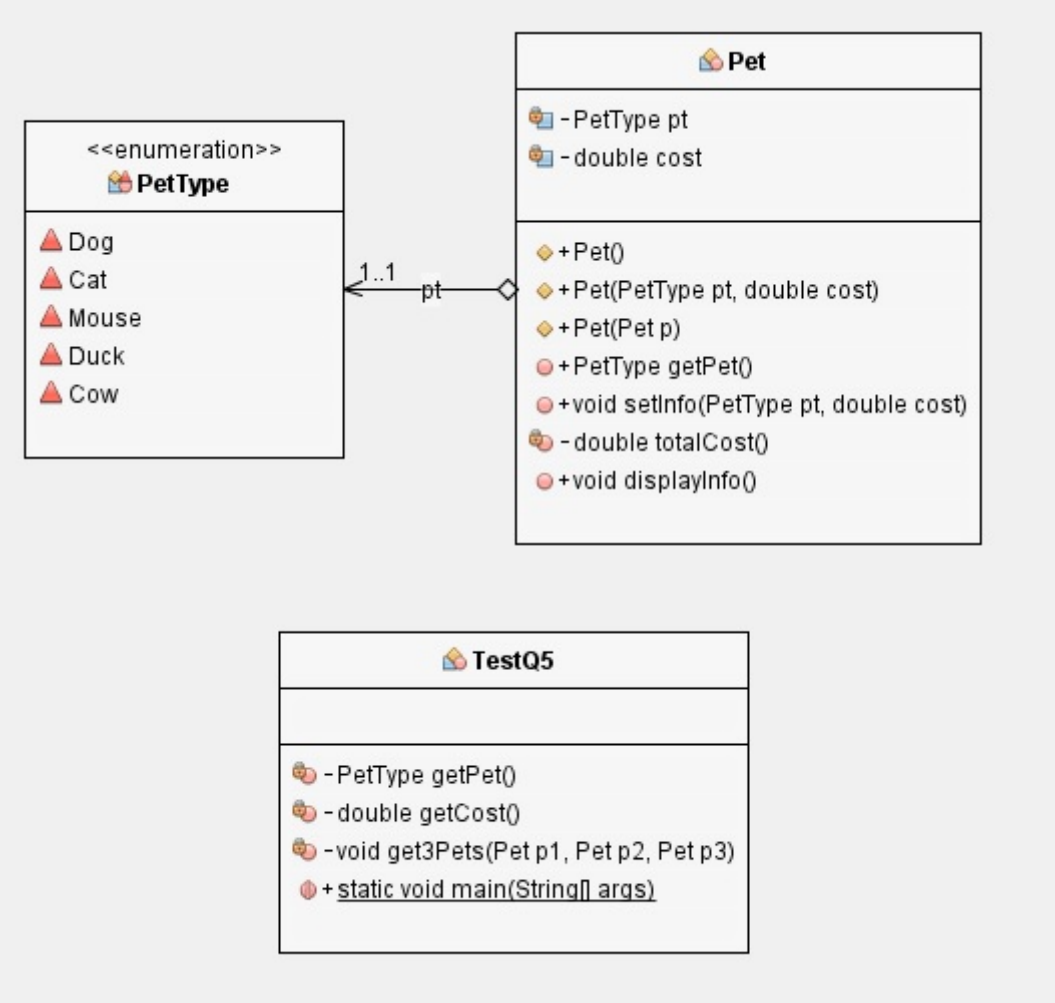
class Quadrilateral (2.5 marks):
Main class (1 marks):
Compilation (0.5 mark):

Question **5**

Not answered

Marked out of 4.00

Study the following UML diagram:




The enum class **PetType** consists of at least 5 different pets (feel free to change)

The **Pet** class consists of  instance variables, default constructor (initialize the instance variable to the  1st animal specified in your enum class, cost is 0.0), other constructor, copy constructor, assessor and mutator methods. In implementation of this class, you should **FULLY** explore use of **"this"** and avoid duplicated statements. In this class you also have a compute of total cost method, just add 7% of GST to the cost. A display info method to display the info of a pet. See below the screen captured for the display format.

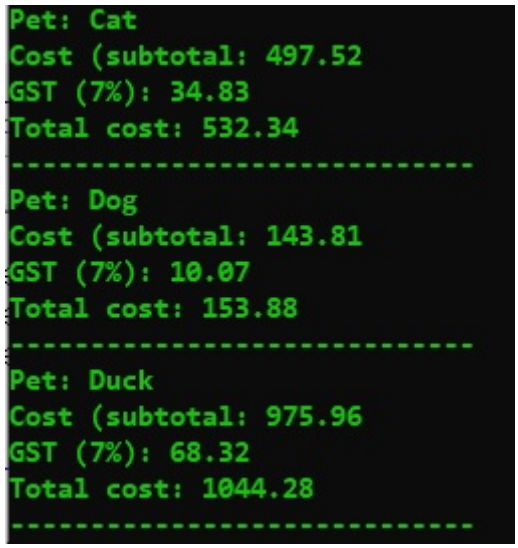
The main class consists of a main method and two user defined methods. Note that the user defined methods are **NON-STATIC**.

- Method **getPet** generates and returns one of the 5 pets specified in the enum class.

- Method **getCost** generates and returns the cost of pet, at least 100.0.

- Method **get3Pets** receives the three **Pet** objects, generates and returns three types of pets (can be the same) via the reference parameters,

- In the main method, you should invoke the **getThreePets** method to have the info of three types of pets, and print out the information (in main) according to the following format:




Write a complete Java program called **YourName_Q5.java** and upload this Java file

**Note that if you have no time to upload the Java program, you can copy and paste your program in the answer box (auto save).**