COMP228 Lab3 - Java programming language

Java Programming (Centennial College)

Centennial College

COMP 228: Java Programming
LAB #3 – Using Inheritance and Polymorphism

Student: _____

Due Date: **Week 7**.
Purpose: The purpose of this Lab assignment is to:
- Practice the use of Inheritance
- Practice the use of Polymorphism.

References: Learning materials for weeks 5 and 6, textbook, and other references (if any)

Be sure to read the following general instructions carefully:
This lab should be completed individually by all the students.

YOU NEED TO SUBMIT THE FOLLOWING 2 DOCUMENTS IN THE DROPBOX TITLED LAB3:
1. THE FIRST ONE IS A WORD DOCUMENT. USE THIS DOCUMENT AND ADD SCREEN SHOTS OF THE RUNNING STATE OF EACH EXERCISE (If there are more than 1 exercise). DO NOT DELETE THE QUESTIONS. THE SCREEN SHOTS SHOULD FOLLOW EACH QUESTION AND COVER ALL THE ASPECTS/FUNCTIONALITIES OF EACH EXERCISE. AFTER THE SCREEN SHOTS PLEASE COPY THE CODE FROM THE CODE WINDOW AND PASTE THE COMPLETE CODE. DO NOT GIVE ME SCREEN SHOTS OF THE CODE. DO NOT ZIP THIS FILE AND KEEP IT SEPARATE FROM YOUR ZIPPED PROGAM FILE.
2. SUBMIT ALSO ONE ZIPPED PROJECT FILE THAT CONTAINS ALL THE EXERISES SEPARATELY INTO THE SAME DROP BOX.

This material provides the necessary information you need to complete the exercises.

You must name your Eclipse project according to the following rule:

YourFullName_COMP228Labnumber
Example: **JohSmith_COMP228Lab3**

Each exercise should be placed in a separate package named *exercise1*, *exercise2*, etc.

Submit your assignment in a **zip file** that is named according to the following rule:
YourLastName_COMP228Labnumber.zip
Example: **JohSmith_COMP228Lab3.zip**

Apply the naming conventions for variables, methods, classes, and packages:

- *variable names* start with a *lowercase* character
- *classes* start with an *uppercase* character
- **packages** use only *lowercase* characters
- *methods* start with a *lowercase* character

Exercise 1

Write a Java application that implements different types of insurance policies for employees of an organization.

Let **Insurance** be an abstract superclass and **Health** and **Life** two of its subclasses that describe respectively health insurance and life insurance.

The **Insurance** class defines an instance variable of type **String** to describe the **type of insurance** and an instance variable of type **double** to hold the **monthly cost** of that insurance.

Implement the **get** methods for both variables of class **Insurance**. Declare also two **abstract** methods named **setInsuranceCost()** and **displayInfo()** for this class.

The **Life** and **Health** class should implement **setInsuranceCost** and **display** methods by setting the appropriate monthly fee and display the information for each insurance type.

Write a driver class to test this hierarchy. This application should ask the user to enter the type of insurance and its monthly fee. Then, will create the appropriate object (Life or Health) and display the insurance information.

As you create each insurance object, place an **Insurance** reference to each new **Insurance** object into an array. Each class has its own **setInsuranceCost** method. Write **a polymorphic** screen manager that walks through the array sending **setInsuranceCost** messages to each object in the array and displaying this information on the screen.

(3 marks)

```
C:\Users\fayab\.jdks\openjdk-17\bin\java.exe
Please select type of Insurance :
  Life
  Health
life
Please enter the cost of Insurance:
900
Please select type of Insurance :
  Life
  Health
Health
Please enter the cost of Insurance:
500
Type of insurance: life
Insurance Cost: $900.0
Type of insurance: health
Insurance Cost: $500.0
```

```java
package com.company;


public abstract class Insurance {
    private String typOfInsurance;
    private double monthlyCost;

    public Insurance(String typOfInsurance, double monthlyCost) {
        this.typOfInsurance = typOfInsurance;
        this.monthlyCost = monthlyCost;
    }

    Insurance[] Objects = new Insurance[]{};

    public String getTypOfInsurance() {
        return typOfInsurance;
    }

    public double getMonthlyCost() {
        return monthlyCost;
    }

    public abstract double setInsuranceCost(double monthlyCost);

    public abstract void displayInfo();


    public String toString() {

        return "Type of insurance: " + typOfInsurance +
                "\nInsurance Cost: $" + monthlyCost;
    }
}
```

```java
package com.company;

public class Health extends Insurance{
    public Health(String typOfInsurance, double monthlyCost) {
        super(typOfInsurance, monthlyCost);
    }

    @Override
    public double setInsuranceCost(double monthlyCost) {
        //monthlyCost=monthlyCost;
        return monthlyCost;
    }

    @Override
    public void displayInfo() {
```

```
        System.out.println("Type of insurance:  "+this.getTypOfInsurance()+"\n
Insurance Cost: "
                +this.getMonthlyCost());
    }
}
```

```
package com.company;

public class Life extends Insurance{
    public Life(String typOfInsurance, double monthlyCost) {

        super(typOfInsurance, monthlyCost);
    }

    @Override
    public double setInsuranceCost(double monthlyCost) {
        monthlyCost=monthlyCost;
        return monthlyCost;
    }

    @Override
    public void displayInfo() {
        System.out.println("Type of insurance:  "+this.getTypOfInsurance()+"\n
Insurance Cost:  "
                +this.getMonthlyCost());
    }
}
```

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

        Insurance[] insurances=new Insurance[3];

        Scanner scanner=new Scanner(System.in);
        String selection ="";
        double enterCost;


        for(int i=0;i<2;i++) {
            System.out.println("Please select type of Insurance :" +
                    "\n  Life  " +
                    "\n  Health ");
            selection = scanner.next().toLowerCase();
            if (selection.equals("life")) {
                System.out.println("Please enter the cost of Insurance:");
                enterCost = scanner.nextDouble();
                Health health = new Health(selection, enterCost);
                insurances[i] = health;
               // health.displayInfo();
//

            } else {
                System.out.println("Please enter the cost of Insurance:");
                enterCost = scanner.nextDouble();
```

```
                Life life = new Life(selection, enterCost);
                insurances[i] = life;
                //life.displayInfo();
            }
        }
        for (Insurance myInsurance:
             insurances) {
            System.out.println(myInsurance);

        }
    }

}
```

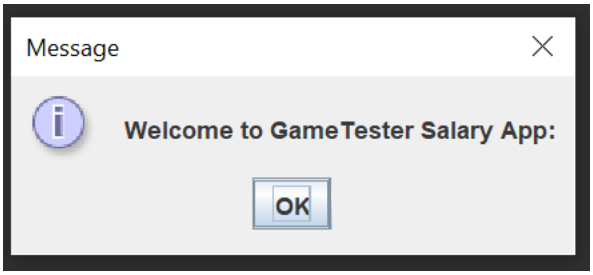
Exercise #2:

Create an abstract class called GameTester. The GameTester class includes a name for the game tester and a boolean value representing the status (full-time, part-time).

Include an abstract method to determine the salary, with full-time game testers getting a base salary of $3000 and part-time game testers getting $20 per hour.

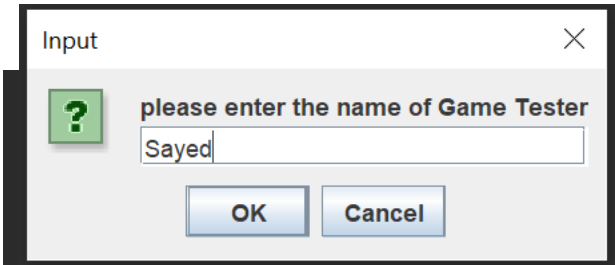
Create two subclasses called FullTimeGameTester, PartTimeGameTester. Create a console application that demonstrates how to create objects of both subclasses. Allow the user to choose game tester type and enter the number of hours for the part-time testers.

(3 marks)




```
C:\Users\fayab\.jdks\openjdk-17\bin\java
Gamer:Sayed
SayedSalary:3000.0

Process finished with exit code 0
```




```
package com.company;

public abstract class GameTester {
    private String name;
    private boolean fulTime;

    public GameTester(String
name,boolean fulTime) {
        this.name = name;
        this.fulTime=fulTime;
    }
public String getName(){
        return name;
```

```
}
    public abstract double determineSalary();
```

```
package com.company;

public class FullTimeGameTester extends GameTester{

private double baseSalary;
    public FullTimeGameTester(String name) {
        super(name,true);
    }

    @Override
    public double determineSalary() {
        baseSalary=3000;
        return baseSalary;
    }
```

```
package com.company;

import javax.swing.*;

public class PartTimeGameTester extends GameTester{

private int hourlyPay;
private double salary;
private double NumberOfHours;
    public PartTimeGameTester(String name,double numberOfHours) {
        super(name,false);
        this.hourlyPay=20;
        this.NumberOfHours=numberOfHours;
    }

    @Override
    public double determineSalary() {
         salary=hourlyPay*NumberOfHours;
         return salary;
    }
}
```

Exercise #3:

CityToronto bank provides mortgages for individuals and businesses up to $300,000. Write a Java application that keeps track of mortgages and computes the total amount owed at any time (mortgage amount + interest).

Design the following classes to implement your application:

Mortgage – an abstract class that implements the *MortgageConstants* interface. A Mortgage includes a mortgage number, customer name, amount of mortgage, interest rate, and term.

Don't allow mortgage amounts over $300,000. Force any mortgage term that is not defined in the *MortgageConstants* interface to a short-term, one year loan. Create a *getMortgageInfo* method to display all the mortgage data.

MortgageConstants – includes constant values for *short-term* (one year), *medium-term* (three years) and *long-term* (5 years) mortgages. It also contains constants for bank name and the maximum mortgage amount.
BusinessMortgage – extends Mortgage. Its constructor sets the interest rate to 1% over the current prime rate.
PersonalMortgage - extends Mortgage. Its constructor sets the interest rate to 2% over the current prime rate.
ProcessMortgage – a main class that create an array of 3 mortgages. Prompt the user for the current interest rate. Then in a loop prompts the user for a mortgage type and all relevant information for that mortgage. Store the created Mortgage objects in the array. When data entry is complete, display all mortgages.

(4 marks)

```
C:\Users\fayab\.jdks\openjdk-17\bin\java.exe
Please select type of mortgage :
 1: Business Mortgage
 0: Personal Mortgage
0
Please enter a number:
7867
Please enter your name:
Sayed
PLease enter amount of loan:
250000
Please enter interest rate:
2
Please enter term:
1
Please select type of mortgage :
 1: Business Mortgage
 0: Personal Mortgage
1
Please enter a number:
5667
Please enter your name:
Fayaz
PLease enter amount of loan:
300000
Please enter interest rate:
3
```

```
Please enter term:
4
Please select type of mortgage :
 1: Business Mortgage
 0: Personal Mortgage
0
Please enter a number:
23456
Please enter your name:
Khaled
PLease enter amount of loan:
120000
Please enter interest rate:
5
Please enter term:
2
Personal Mortgage:  Mortgage number:7867 Amount of loan:$250000.0 Customer Name:Sayed Interest Rate:2.0 Term of Loan:1
Business Mortgage:  Mortgage number:5667 Amount of loan:$300000.0 Customer Name:Fayaz Interest Rate:3.0 Term of Loan:4
Personal Mortgage:  Mortgage number:23456 Amount of loan:$120000.0 Customer Name:Khaled Interest Rate:5.0 Term of Loan:2

Process finished with exit code 0
```

```java
package com.company;

public abstract class Mortgage implements MortgageConstant {
    private int mortgageNumber;
    private String customerName;
    private double amountOfMortgage;
    private double interestRate;
    private int term;
    public Mortgage(int mortgageNumber,
                    String customerName,
                    double amountOfMortgage,
                    double interestRate,int term){
        this.mortgageNumber=mortgageNumber;
        this.customerName=customerName;
        this.amountOfMortgage=amountOfMortgage;
        this.interestRate=interestRate;
        this.term=term;
    }
    public int getMortgageNumber(){
        return mortgageNumber;
    }
    public void setMortgageNumber(){
        this.mortgageNumber=mortgageNumber;
```

```
    }
    public String getCustomerName(){
        return customerName;
    }
    public void setCustomerName(){
        this.customerName=customerName;
    }
    public double getAmountOfMortgage(){
        return amountOfMortgage;
    }
    public void setAmountOfMortgage(double interestRate){
        if (amountOfMortgage>MAXIMUMLOANAMOUTN){
            System.out.println("Mortgage amount should not be greater than
$300,000");
        }
        else {
            this.amountOfMortgage=amountOfMortgage;
        }
    }

    public double getInterestRate(){
        return interestRate;
    }
    public void setInterestRate(){
        this.interestRate=interestRate;
    }
    public int getTerm(){
        return term;
    }
    public void setTerm(){
        if(term<3)
            this.term= SHORTTERM;
        else if (term==3)
            this.term=MEDTERM;
        else
            this.term=LONGTERM;
    }



    public String getMortgageInfo(){
        return "[mortgageNumber =" + mortgageNumber +" "+
                ",customerName=" + customerName +" "+
                ", amount=" + amountOfMortgage +" "+
                ", interestRate=" + interestRate +" "+
                ", term=" + term + "]";
    }

    @Override
    public String toString(){

        return "Mortgage number: "+ mortgageNumber +" "+
                "Amount of loan: "+amountOfMortgage+" "+
                "Customer Name: " + customerName +" "+
                "Interest Rate: " + interestRate+" "+
                "Term of Loan in years:"+ term;
```

```
    }
}
```

```
package com.company;

public interface MortgageConstant {
    final int   SHORTTERM = 1;
    final  int MEDTERM=3;
    final int LONGTERM=5;
    final int MAXIMUMLOANAMOUTN=300000;

    final String BANKNAME="TorontoCityBank";

}
```

```
package com.company;

public class PersonalMortgage extends Mortgage {


    public PersonalMortgage(int mortgageNumber,
                            String customerName,
                            double amountOfMortgage,
                            double interestRate, int term) {
        super(mortgageNumber, customerName, amountOfMortgage, interestRate,
term);
        setAmountOfMortgage(getInterestRate()+2);

    }

    @Override
    public String toString() {
        return "Personal Mortgage:  Mortgage number:" +
this.getMortgageNumber() +" "+
                "Amount of loan:$" + this.getAmountOfMortgage() +" "+
                "Customer Name:" + this.getCustomerName() +" "+
                "Interest Rate:" + this.getInterestRate() +" "+
                "Term of Loan:" + this.getTerm();

    }

}
```

```
package com.company;

public class BusinessMortgage extends Mortgage {


    public BusinessMortgage(int mortgageNumber,
                            String customerName,
                            double amountOfMortgage,
```

```
                            double interestRate, int term) {
        super(mortgageNumber, customerName, amountOfMortgage, interestRate,
term);
        setAmountOfMortgage(getAmountOfMortgage() + 1);
        ;
    }

    @Override
    public String toString() {
        return "Business Mortgage:  Mortgage number:" +
this.getMortgageNumber() +" "+
                "Amount of loan:$" + this.getAmountOfMortgage() +" "+
                "Customer Name:" + this.getCustomerName() +" "+
                "Interest Rate:" + this.getInterestRate() +" "+
                "Term of Loan:" + this.getTerm();

    }
}
```

```
package com.company;

import java.util.Scanner;

public class MortgageProcessor {

    public static void main(String[] args) {

        Mortgage[] mortgages=new Mortgage[3];
        Scanner scanner=new Scanner(System.in);
        int selection =0;
        int mortgageNumber;
        String name;
        double amountOfLoan;
        double interestRate;
        int term;

        for(int i=0;i<3;i++){
            System.out.println("Please select type of mortgage :" +
                    "\n 1: Business Mortgage " +
                    "\n 0: Personal Mortgage");
            selection=scanner.nextInt();
            if(selection==1){
                System.out.println("Please enter a number:");
                mortgageNumber=scanner.nextInt();

                System.out.println("Please enter your name:");
                name=scanner.next();

                System.out.println("PLease enter amount of loan:");
                amountOfLoan=scanner.nextDouble();

                System.out.println("Please enter interest rate:");
                interestRate=scanner.nextDouble();
```

```
                System.out.println("Please enter term:");
                term=scanner.nextInt();

                BusinessMortgage businessMortgage= new
BusinessMortgage(mortgageNumber,name,
                        amountOfLoan,interestRate,term);
                mortgages[i]=businessMortgage;


            }
            else {

                System.out.println("Please enter a number:");
                mortgageNumber=scanner.nextInt();

                System.out.println("Please enter your name:");
                name=scanner.next();

                System.out.println("PLease enter amount of loan:");
                amountOfLoan=scanner.nextDouble();

                System.out.println("Please enter interest rate:");
                interestRate=scanner.nextDouble();

                System.out.println("Please enter term:");
                term=scanner.nextInt();
                PersonalMortgage PersonalMortgage= new
PersonalMortgage(mortgageNumber,name,
                        amountOfLoan,interestRate,term);
                mortgages[i]=PersonalMortgage;

            }


        }

        for (Mortgage myMortgage:mortgages
             ) {
            System.out.println(myMortgage);
        }
    }
}
```

Evaluation:

Functionality	
Correct implementation of classes (instance variable declarations, constructors, getter and setter methods, etc.)	40%
Correct implementation of driver classes (declaring and creating objects, calling their methods, interacting with user, displaying results)	40%
Comments, correct naming of variables, methods, classes, etc.	5%
Friendly input/output	15%
Total	100%