Piyush Vishnoi

Rollno- R171218071

Sapid: 500067083

Q.1 Create the abstract class Animal and a set of two or three animal sub-classes such as Tiger and Lion. Details for Animal class is- o String getName() - which returns the name of the animal. o int getAge() - which returns the age in years. o void talk() - which causes the animal to display on the screen their name, age and what type of animal they are.

Create a Zoo class containing an array of references to animals with the functionality below-

The Zoo provides the following methods :- o addAnimal(Animal newAnimal) - which adds a new animal to the Zoo. o displayAnimals() - which displays a list of the type and name of all animals

in the zoo. o void feedingTime() - which causes all animals in the zoo to talk one after the other.

The main class has following functionalities- Add a new animal to the zoo - first prompting for name and type and creating the animal. Display all animals currently in the zoo. Display the name and age of the animal in a selected type. Trigger feeding time. Implement a menu-driven main program which uses the Zoo and the various animal classes and test it.

Code /*

* To change this license header, choose License Headers in Project Properties. * To change this template file, choose Tools | Templates

* and open the template in the editor.

```
*/

package practice_java;

/** *

* @author Piyush

*/
import java.util.ArrayList;

import java.util.Collections;




abstract class Animal

{

}

{
```

```
abstract String getName();

abstract int getAge();




class Tiger extends Animal




private String name; // name of Tiger.

private int age; // age of Tiger.

// Constructor

public Tiger(String name, int age)




{
```

```
this.name = name;









}









this.age = age;









public String getName()

{

}

//method required to extend the animal class.

public int getAge()

{
```

```
return age;

}




return name;









}

class Lion extends Animal

{




private String name; // name of Lion.

private int age; // age of Tiger.

// Constructor
```

```
public Lion(String name, int age)

{

}

this.name = name;

this.age = age;

public String getName()

{

}
```

```
return name;




public int getAge()

{

return age;

}

}

class Cheeta extends Animal

{




private String name; // To hold name of Cheeta.

private int age; // To hold age of Tiger.
```

```
// Constructor that accept name as parameter.

public Cheeta(String name, int age)

{

}




this.name = name;

this.age = age;




public String getName()

{

}
```

```
//method required to extend the animal class.




return name;




public int getAge()

{

return age;

}

}




public class Animal_zoo

{
```

```
{



public static void main(String[] args)



// create an arrayList of animals

ArrayList<Animal> animals = new ArrayList<Animal>();

animals.add(new Tiger("tiger", 10));



animals.add(new Lion("lion", 15));

animals.add(new Lion("elephant", 13));

animals.add(new Tiger("zebra", 12));

animals.add(new Cheeta("monkey", 18));
```

```
for(Animal animal : animals)

{

}

}

}
```

```
System.out.println(animal.getName()+ " : " + animal.getAge());
```

Output

Q.2 Create a class Rectangle. The class has attributes length and width, each of which defaults to 1. It has member functions that calculate the perimeter and the area of the rectangle. It has set and get functions for both length and width. The set functions should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0 otherwise should throw a user defined exception. Create a more sophisticated constructor Rectangle of the class. This class stores only the Cartesian coordinates of the four corners of the rectangle. The constructor calls a set function that accepts four sets of coordinates and verifies that each of these is in the first quadrant with no single x or y coordinate larger than 20.0 otherwise throw an exception. The set function also verifies that the supplied coordinates do, in fact, specify a rectangle otherwise throw an exception.

Member functions calculate the length, width, perimeter and area. The length is the larger of the two dimensions. Include a function square that determines if the rectangle is a square.

Code

```
package practice_java;

import java.util.*;

import java.lang.*;

//custom exception

class OutOfRange extends Exception {



Double a;









}









OutOfRange(Double b) { a=b;



}

public String toString(){



return ("Out Of Range Value provided "+a) ; }
```

```
//class rectangle

public class Rectangle

{ Scanner sc=new Scanner(System.in);


Double length=1, width=1; private String area; private String perimeter;


public Double getLength() {


return this.length;


}

public Double setLength(Double length) {//checking condition


try {

if(length < 0.0 || length > 20.0)




{


} }
```

```
throw new OutOfRange(length);//exception throw




catch (OutOfRange e) {

System.out.println(e);









}




return this.length = length;

}
public Double getWidth() {

return this.width;

}
public Double setWidth(Double width) {
```

```
try {
if(width < 0.0 || width > 20.0)




{


} }



throw new OutOfRange(width);




catch (OutOfRange e) {


System.out.println(e);


}


return this.width = width;


}
public double area()//area calculation
```

```
{
return (this.length*this.width);

}
public double perimeter() {




}




return (2*(this.length+this.width));




public Rectangle(double x1, double x2, double x3, double x4, double y1, double y2, double
y3, double y4)




{
```

```
}

if(Math.hypot(x1-x2, y1-y2)==Math.hypot(x3-x4, y3-y4)) {

System.out.println("Rectangle is valid ");

}
else if(x1==x3 && x3==x4 && y1==y2 && y3==y4) {

System.out.println("It is square");

} else

this.setLength(Math.sqrt((y4 - y1) * (y4 - y1) + (x4 - x1) * (x4 - x1)));
this.setWidth(Math.sqrt((y2 - y1) * (y2 - y1) + (x2 - x1) * (x2 - x1)));
System.out.println("Length is "+getLength()); System.out.println("Width is" +getWidth());

System.out.println("Rectangle is invalid ");
```

public static void main(String args[]) {

}

output

Rectangle obj=new Rectangle(3.0,5.0,4.0,5.0,3.0,7.0,4.0,7.0); System.out.println("area is" +obj.area()); System.out.println("perimeter is" +obj.perimeter());

}

Q.3 Imagine a publishing company that markets both book and audio cassette versions of its works. Create a class publication that stores the title (a string) and price (type float) of a publication. From this class derive two classes: book, which adds a page count (type int), and tape, which adds a playing time in minutes (type float). Each of these three classes should have a proper constructors, destructors and display functions to display its data. Write a main() program to test (a) order of constructor and destructor invocation, (b) the book and tape classes by creating instances of them, asking the user to fill in data and then displaying the data. Start with the publication, book, and tape classes of ‘Ques-1’. Add an interface sales that declares functions to hold sales. Include a function to get sales amounts from the user, and a function to display the sales figures. Alter the book and tape classes so they are derived from both publication and sales. An object of class book or tape should

input and output sales data along with its other data. Write a main() class to create a book object and a tape object and exercise their input/output capabilities.

Code /*

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates * and open the template in the editor.

*/

```
package practice_java;

/** *

* @author Piyush */

import java.util.*;//interface interface sales{

int get_sales(int amount); void display_sales();

}

class Publication//class
```

```
{
String t;float p;
Publication(String tittle,float price) {
t=tittle;
p=price;
}
void display()//display method
{
System.out.println("tittle is "+t); System.out.println("price is Rs."+p); }

}
```

```
abstract class book extends Publication implements sales {
int c,a;
book(int count)//page count

{ super("c",24.8f); c=count;}

void display()
{System.out.println("no of pages added are "+c ); }

public int get_sales(int amount)
```

```
{ a=amount; return a;

}

} }

public void display_sales() {

System.out.println("the sales amount is Rs."+a);

abstract class tape extends Publication implements sales//making it abstract to extend and implement

{

float m; tape(float min) {

super("t",42.8f);

m=min;} void display()

{System.out.println("playing time in minutes is "+m+ " min");}
```

```
}

public class Companyprogram {

public static void main(String args[])

{

Scanner sc= new Scanner(System.in); System.out.println("Enter the tittle"); String
tittle=sc.nextLine();

System.out.println("enter the price"); float price=sc.nextFloat();

System.out.println("enter the page count"); int count=sc.nextInt();

System.out.println("enter the minute");

float min=sc.nextFloat(); System.out.println("enter the sale amountt"); int
amount=sc.nextInt();
```

```
Publication obj=new Publication(tittle,price); obj.display();


book ob=new book(count) {}; ob.display(); ob.get_sales(amount); ob.display_sales();


tape o=new tape(min) { @Override


public int get_sales(int amount) {


throw new UnsupportedOperationException("Not supported yet."); //To change body of
generated methods, choose Tools | Templates.









}









@Override

public void display_sales() {
```

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

} };

o.display();

}

} Output

Q.4 Assume that a bank maintains two kinds of accounts for customers, one called as savings account and other as current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class account that stores customer name, account number (Should generate automatically if new customer added) and type of account. From this derive the classes cur_acct and sav_acct to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks: a. Accept deposit from a customer and update the balance b. Display the balance. c. Compute and deposit interest. d. Permit withdrawal and update the balance. e. Check for minimum balance, impose penalty, necessary and update the balance

Code

```
import

java.util.Scanner;

public class account { //make a super class account

int balance =500;//declare min balance

int rate=4;//declare intrest rate for compound intrest

        public static void main(String[] args) {

        sav_acct obj=new sav_acct();// make object of saving account cur_acct obj1=new
        cur_acct();//make object of current account

        Scanner sc=new Scanner(System.in); System.out.println("enter your name");

        String customer_name =sc.nextLine(); System.out.println("enter your account number");
```

```
int account_number =sc.nextInt();

System.out.println("enter type of account");

System.out.println("types S for saving account and C for current account"); char
type=sc.next().charAt(0);

boolean q=true;//condition to return back to the menu

while(q){

if(type=='S'){

System.out.println("1.deposit"); System.out.println("2.withdraw");
System.out.println("3.compound intrest"); System.out.println("4.EXIT");

System.out.println("enter any no. to perform the task"); int n=sc.nextInt();

switch(n){ case 1:

obj.deposit();

break ; case 2:

obj.withdraw();
```

```
break; case 3:

obj.compound(); case 4:

System.exit(0); default:

System.out.println("invalid");

} }

else{
System.out.println("1.display balance");

System.out.println("2.EXIT");
System.out.println("enter number to perform the following task");

int c=sc.nextInt(); if(c==1)

obj1.display();
```

```
else{ System.exit(0);

} }}}}

class cur_acct extends account{ //current account class public void display(){

if(balance<500){

System.out.println("not enough money deposit the minimum balance");

}

else{

System.out.println("your balance is Rs."+ balance);

}
```

```
} }


class sav_acct extends account{// saving acount class Scanner sc1=new
Scanner(System.in);


public void deposit(){


System.out.println("enter the amount you want to deposit");

int amt=sc1.nextInt();

balance=balance+amt;

System.out.println("Rs."+ amt +" have been deposited to your account");
System.out.println("updated balance is Rs." +balance);









}

public void withdraw(){


System.out.println("enter the amount you want to withdraw"); int amt1=sc1.nextInt();

if(balance<500){


System.out.println("not sufficient balance deposit the min balance"); }


else{

balance=balance-amt1;

System.out.println( "Rs."+ amt1 +" have been withdrawn from your account ");
```

```
System.out.println("updated balance is Rs."+balance); }}

public void compound(){ int r=1;

System.out.println("enter the no. of years for intrest applied"); int t=sc1.nextInt();

for(int i=1;i<=t;i++){ r=r*(1+rate);
```