LAB PROGRAM 4

(*) 2-1-24

2/1/2024

Develop a Java program to create an abstract class name shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class shape. Each one of the classes contain only the method printArea() that prints the area of the given shape.

```
import java.util.*;
class InputScanner {
    Scanner s1 = new Scanner(System.in);
}

abstract class shape extends InputScanner {
    double a;
    double b;

    abstract void getInput();
    abstract void displayArea();
}
```

```
class Rectangle extends Shape {
    void getInput() {
        System.out.println("enter value of a");
        a = s1.nextDouble();
        System.out.println("enter value of b");
        b = s1.nextDouble();
    }
    void displayArea() {
        System.out.println("area of rectangle
                is" + (a * b));
    }
}

class Triangle extends Shape {
    void getInput() {
        System.out.println("Enter the value of a");
        a = s1.nextDouble();
        System.out.println("enter the Value of b");
        b = s1.nextDouble();
    }
```

```
Void displayArea () {
    System.out.println ("area of Triangle is"
                        + (a*b/2));
    }
}

class Circle extends Shape {
    void getInput () {
        System.out.println ("Enter value of b:");
        b=s1.nextDouble ();
    }
    void displayArea () {
        System.out.println ("area of circle is"
                            + (b*b*3.14)); }
}

class AbstractDemo {
    public static void main (String args[]) {
        Rectangle r= new Rectangle ();
        Circle c = new Circle ();
        Triangle t = new Triangle ();
```

```
r. getInput();
r. displayArea();

c. getInput();
s. displayArea();

t. getInput();
t. displayArea();
```

System.out.println("name: Sohan T Sanjeev");

System.out.println("USN: 2023BMS02T32");

}

}

Output:-

```
enter the value of a:
1
enter the value of b:
4
area of rectangle is 4.0
enter the value of b:
3
area of circle is 28.26
enter value of a
5
enter value of b
7
area of triangle is 17.5
```

name : Sohan T Sanjeev

USN : 2023BMS02932

Lab-5

Develop a Java program to create a class Bank that contains two kinds of account for its customers, one called savings account and other current account provides cheque book facility. The current account holders should 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 and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements include necessary methods in order to achieve the following tasks.

(•) accept deposit from customer and update the balance

(•) Display the balance

(•) Compute and deposit interest

(•) Permit withdrawal and update balance

(•) Check for minimum balance, impose penalty if necessary and update balance

```
import java.util.*;
class Account {
String name;
int accno;
String type;
double balance;
Account(String name, int accno, String type, double
                                    balance) {
  this.name = name;
  this.accno = accno;
  this.type = type;
  this.balance = balance;
}
void deposit(int amount)
  balance += amount;
}
void withdraw(double amount){
if((balance - amount) >= 0) {
  balance -= amount;
}
else {
System.out.println("insufficient balance");
}
}
```

```
void display () {
 System out println ("Name : " + name + "\n"
   + "Account No : " + accno + "\n" + "Type : "
   + type + "\n" + "balance : " + balance + "\n");
 }
}

class Saving Account extends Account {
 Private static int rate = 5;
Current Saving Account (String name, int accno,
 String type, double balance) {
    Super (name, accno, type, balance);
}
 void balance with Interest () {
    balance += balance * rate / 100;
System out println ("balance : " + balance);
}
}

public class main {

 public static void main (String args[]) {
    Scanner s = new Scanner (System. in);
    System out println (" Enter the account type
       (current or deposit)");
     String type = s.next ();
```

```
System.out.println("Enter account number");
int accno = S.nextInt();

System.out.println("Enter initial balance:");
double balance = S.nextDouble();

Account acc = new Account(name, accno, type,
              balance);
Saving Account sa = new Saving Account(name,
              accno, type, balance);
double amount;
while (true){
  if(acc.type.equals("savings")){
    System.out.println("\n --- MENU --\n");
    System.out.println("1: Deposit \n 2. Withdra
    \n 3. Compute Interest for Savings Accoun
    \n 4. Display Account details \n 5. exit \n
  System.out.println("Enter your choice");
  int choice = S.nextInt();

  Switch(choice){
    case 1: System.out.println("Enter deposit
                amount");
```

```
amount = s.nextDouble();
sa.deposit(amount);
break;

case 2: System.out.println("Enter withdrawal
        amount");
amount = s.nextDouble();
sa.withdraw(amount);
break;

case 3: sa.balanceWithInterest();
break;

case 4: System.out.println("Details");
sa.display();
break;

case 5: return;

default: System.out.println("Invalid choice");
}

}
```

9/11/24

Lab-6

1) Demonstrate String constructors

2) Demonstrate string length, string literal, concat

```
class SubstringCons {

    public static void main (String args[]) {

        String s1 = new String ();

        String s2 = new String ("hello");

        System.out.println ("s1 = " + s2);

    char chars[] = {'a', 'b', 'c', 'd', 'e'};

    String s3 = new String (chars, 0, 3);

    System.out.println ("s3: " + s3);

    byte ascii[] = {65, 66, 67, 68, 69, 70};

    String s4 = new String (ascii);

    System.out.println (s4);

    String s5 = new String (ascii, 2, 3);

    System.out.println (s5);

    String s6 = "hello";

    System.out.println ("length of string s:"
      + s6.length ());

    String s7 = "java";
```

```
String obj = new String (s6 + s7);
System.out.println ("obj = " + obj);
}
}
```

Output:-

s1 = hello

s3 = abc

ABCDEF

CDE

length of String s6 : 5

obj = hello java

Program 2:-

WAP to create an abstract class Bird with methods fly() & makeSound(). Create Subclass Eagle and Hawk that Extend class & Implement the respective methods to describe how each bird flies & makes a sound

```
abstract class Bird {
    abstract void fly();
    abstract void makeSound();
}
class Eagle Extends Bird {
  void fly() {
      System.out.println("eagle soars high
        in the sky");
  }
void makeSound() {
      System.out.println("eagle makes screechi
        sound");
  }
}
class Hawk extends Bird {
  void fly() {
      System.out.println("hawk glides
        gracefully in air");
  }
void makeSound() {
```

```
System.out.println(" hawk makes distinct cry");
    }
}

class AbstractMain {

    public static void main(String args[]){

        Eagle e = new Eagle();
        Hawk h = new Hawk();
        e.fly();
        e.makeSound();
        h.fly();
        h.makeSound();

}
```

16/11/24.

Output:-

Eagle soars high in the sky.
Eagle makes screeching sound
hawk glides gracefully in air
hawk makes distinct cry.

(3) WAP to create a generic class stack which hold 5 integers and 5 doubles.

```
public class GenericStack<T> {
    private Object[] stackArray;
    private int top;
    private static final int MAX_SIZE = 10;

public GenericStack() {
    stackArray = new Object[MAX_SIZE];
    top = -1;
}

public void push(T element) {
    if (top < MAX_SIZE - 1) {
        stackArray[++top] = element;
    }
    else {
        System.out.println("Stack is full. Cannot
            push more elements");
    }
}
```

```
public T pop () {
    if (!isEmpty()) {
        @SuppressWarnings ("unchecked")
        T element = (T) stackArray[top--];
        System.out.println ("popped: "+element);
        return element;
    }
    else {
        System.out.println ("Stack is Empty.
        Cannot pop elements");
        return null;
    }
}

Public boolean isEmpty () {
    return top == -1;
}

Public static void main (String args[]) {
    GenericStack <Integer> integerStack = new
        GenericStack <> ();
    integerStack.push (1);
    integerStack.push (2);
```

```
IntegerStack (3);
IntegerStack.pop ();
```

```
GenericStack<Double> doubleStack = new Gen
        Stack<> ();
doubleStack.push(1.5);
doubleStack.push(2.5);
doubleStack.push(3.5);
doubleStack.pop();
  }
}
```

12/11/24

O/P:

```
pushed 1
pushed 2
pushed 3
popped 3
pushed 1.5
pushed 2.5
pushed 3.5
Popped 3.5
```