Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Academic Year: 2022-2023

Name – Aksh Dilip Nishar **SAP ID - 60004220123**

Experiment No - 6

AIM: To implement Functions, recursive functions, andOverloading

THEORY:

This is a Java Program to Find Area of Square And Rectangle using MethodOverloading.
We declare three methods of same name but with different number of arguments orwith different data types. Now when we call these methods using objects, corresponding methods will be called as per the number of arguments or their datatypes.Here is the source code of the Java Program to Find Area of Square, Rectangle using Method Overloading. The Java program is successfully compiled and run on a Windows system. The program output is also shown below.
Overloading allows different methods to have the same name, but different signatures where the signature can differ by the number of input parameters or type of input parameters or both.Overloading is related to compile-time (or static) polymorphism.

CODE:

```
class Overload
{
    void area(float x)
    {
       System.out.println("Aksh Nishar 60004220123"); System.out.println("the
        area of the square is "+Math.pow(x, 2)+" sq units");
    }
    void area(float x, float y)
    {
        System.out.println("the area of the rectangle is "+x*y+" sq units");
    }
}
class Main
{
     public static void main(String args[])
          {
            Overload ob = new Overload();ob.area(5);
            ob.area(11,12);
        }
}
```

OUTPUT:

```
D:\java>javac djsce.java

D:\java>java djsce
Aksh Nishar 60004220123
the area of the square is 25.0 sq units
the area of the rectangle is 132.0 sq units

D:\java>
```

Write menu driven program to implement recursive functions for followingtasks.

Theory:

In this program we will be writing a menu driven program implementing the concept of recursive functions Recursion is the technique of making a function call itself. It is a process in which a method calls itself continuously. A method in java that calls itself is called recursive method. This technique provides a way to break complicatedproblems down into simple problems which are easier to solve. Here we've created multiple classes which execute a given respective function and all traversed using a menu written using switch case.
Recursion in java is a process in which a method calls itself continuously. A
method in java that calls itself is called recursivemethod.
It makes the code compact but complex to understand.Syntax:
returntype methodname(){
//code to be executed methodname();//calling same
method
}

Code:

```
import java.util.Scanner;
public class Main6_2{
// gcd lcm
static public int gcd(int a, int b) {
if (b == 0) {
return a;
}
return gcd(b, a % b);
}
static int lcm(int a, int b) {
return (a / gcd(a, b)) * b;
}
static int dig = 0;
static int s = 0;
// reverse
static int solve(int n) {
if (n != 0) {
dig = dig * 10 + n % 10;
solve(n / 10);
}
return dig;
}
// sum of digits
static int sum(int n) {
if (n != 0) {
s = s + n % 10;
sum(n / 10);
}
return s;
}
static int sum = 0;
static int solveN(int nth) {
if (nth > 0) {
sum = sum + nth;
solveN(nth - 1);
}
return sum;
}
static int a = 0;
static int b = 1;
static int c = 0;
static int m = 1;
```

```
static void fibo(int n) {
if (n > 0) {
c = a + b;
System.out.print(c + " ");
a = b;
b = c;
fibo(n - 1);
}
}
static int multi(int x, int y) {
if (y > 0) {
m = m * x;
multi(x, y - 1);
}
return m;
}
public static void main(String[] args) {
System.out.println("Aksh Nishar 60004220116");
Scanner input = new Scanner(System.in);
System.out.println("Enter the function");
System.out.println("1) To find GCD and LCM");
System.out.println("2) To find X^Y ");
System.out.println("3) To print n Fibonacci numbers");
System.out.println("4) To find reverse of number ");
System.out.println("5) To 1+2+3+4+..+ (n-1)+n");
System.out.println("6) Calculate the sum of digits of a number ");
int fn = input.nextInt();
switch (fn) {
case 1:
int temp;
System.out.println("Enter the two numbers:");
int a = input.nextInt();
int b = input.nextInt();
if (a < b) {
temp = a;
a = b;
b = temp;
}
System.out.print("GCD is :");
System.out.println(gcd(a,  b));
System.out.print("LCM is :");
System.out.println(lcm(a, b));
```

```
break;
case 2:
System.out.println("Enter the x and y value");
int x = input.nextInt();
int y = input.nextInt();
if (y == 0) {
System.out.println("Answer:" + 1);
} else {
System.out.println("Answer:" + multi(x, y));
}
break;
case 3:
// fibo
a = 0;
b = 1;
System.out.println("Enter the nth value");
int count = input.nextInt();
System.out.print(a + " ");
System.out.print(b + " ");
fibo(count - 2);
break;
case 4:
System.out.println("Enter the number");
int n = input.nextInt();
System.out.println("reverse is " + solve(n));
break;
case 5:
System.out.println("Enter the nth value");
int nth = input.nextInt();
for (int i = 1; i <= nth; i++) {
if (i == nth) {
System.out.print(i);
} else {
System.out.print(i + "+");
}
}
System.out.print("=" + solveN(nth));
break;
case 6:
System.out.println("Enter the number");
int m = input.nextInt();
System.out.println("Sum is " + sum(m));
```

```
break;
default:
System.out.println("Invalid input");
}
}
}
```

Output:

```
D:\java>javac djsce.java

D:\java>java djsce
Aksh Nishar 60004220123
Enter the function
1) To find GCD and LCM
2) To find X^Y
3) To print n Fibonacci numbers
4) To find reverse of number
5) To 1+2+3+4+..+ (n-1)+n
6) Calculate the sum of digits of a number
1
Enter the two numbers:
25 5
GCD is :5
LCM is :25
```

```
D:\java>java djsce
Aksh Nishar 60004220123
Enter the function
1) To find GCD and LCM
2) To find X^Y
3) To print n Fibonacci numbers
4) To find reverse of number
5) To 1+2+3+4+..+ (n-1)+n
6) Calculate the sum of digits of a number
6
Enter the number
5555
Sum is 20
```

CONCLUSION: Thus, we have implemented Functions, recursive functions, andOverloading in Java.

Name – Aksh Dilip Nishar **SAP ID - 60004220123**

Experiment No - 7

AIM: To implement Array of Objects

Theory:

In this program we implement the use of array of object, The array of Objects the name itself suggests that it stores an array of objects. Unlike the traditional array stores values like String, integer, Boolean, etc an Array of Objects stores objects thatmean objects are stored as elements of an array, here we are accepting and storing and displaying multiple student data such as name, rollno and marks into its respective object stored in an array. The objects in the array are traversed with each object accepting and displaying the user input it had stored and also sort the data in descending order of the data.

Code:

```
import java.util.Scanner;
class Student
{
int roll,phy,chem,math,total;
String name;
void input()
{
Scanner scan=new Scanner(System.in);
System.out.println();
System.out.print("Enter student name:");
name=scan.nextLine();
System.out.print("Enter Roll_no:");
roll=scan.nextInt();
System.out.println("Enter Marks:");
System.out.print("Physics Marks:");
phy=scan.nextInt();
System.out.print("Chemistry Marks:");
chem=scan.nextInt();
System.out.print("Mathematics Marks:");
math=scan.nextInt();
total=phy+chem+math;
System.out.println();
System.out.println("*****Student details registered*****");

}
void output()
{
System.out.println("Student: "+name+" ,roll_no: "+roll+" ,marks: ");
System.out.println("Physics:"+phy);
```

```
System.out.println("Chemistry:"+chem);
System.out.println("Mathematics:"+math);
System.out.println("Total:"+total);
}
}
class Main7 {
public static void main(String args[])
{
System.out.println("Aksh Nishar 60004220123");int
i,j;
Student s[]=new Student[5]; for(i=0;i<5;i++)
{
s[i]=new Student();
}
System.out.println("Enter Details: ");
for(i=0;i<3;i++)
{
s[i].input();
}
for(i=0;i<3;i++)
{
s[i].output();
}
Student temp;
for(i=0;i<4;i++)
{
for(j=0;j<4-i;j++)
{
if(s[j].total<s[j+1].total)
{
temp=s[j]; s[j]=s[j+1]; s[j+1]=temp;
}
}
}
System.out.println("Student Marks in Descending Order:");
for(i=0;i<5;i++)
{
System.out.println("Student Name: "+s[i].name+", Student RollNo: "+s[i].roll+", Total:"+s[i].total);
}
}
}
```

Output:

```
*****Student details registered*****

Enter student name:sakshi
Enter Roll_no:17
Enter Marks:
Physics Marks:99
Chemistry Marks:98
Mathematics Marks:97

*****Student details registered*****
Student: Sakshi ,roll_no: 872 ,marks:
Physics:67
Chemistry:29
Mathematics:10
Total:106
Student: Parth ,roll_no: 16 ,marks:
Physics:19
Chemistry:21
Mathematics:0
Total:40
Student: sakshi ,roll_no: 17 ,marks:
Physics:99
Chemistry:98
Mathematics:97
Total:294
Student Marks in Descending Order:
Student Name: sakshi, Student RollNo: 17, Total:294
Student Name: Sakshi, Student RollNo: 872, Total:106
Student Name: Parth, Student RollNo: 16, Total:40
Student Name: null, Student RollNo: 0, Total:0
Student Name: null, Student RollNo: 0, Total:0
PS D:\java>
```

CONCLUSION: Thus, we have implemented array of objects in java.

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Academic Year: 2022-2023

Name – Aksh Dilip Nishar **SAP ID - 60004220123**

Experiment No - 8

AIM: To implement Constructors and overloading

WAP find area of square and rectangle using overloaded constructorTheory:

In this program we have implemented the concept of constructor overloading, The constructor overloading can be defined as the concept of having more than one constructor with different parameters so that every constructor can perform a different task. Here we have created a class named Shape and two constructors buthaving different numbers and types of parameters hence applying the concept of Constructor Overloading.
The constructor overloading can be defined as the concept of having more than one constructor with different parameters sothat every constructor can perform a different task.

Code:

```
import java.util.*;
class Shape{
int s1,s2; Shape(int s){
s1=s; s2=s;
}
Shape(int l ,int b){ s1=l;
s2=b;
}
float area(){ return s1*s2;
}
}
public class Main8 {
public static void main(String args[]){
System.out.println("Prayag Mitaliya 60004220259");
Shape square=new Shape(6);
Shape rect=new Shape(4,5);
System.out.println("Area of Square of side 6 is " +square.area());
System.out.println("Area of Rectangle of length 4 and breadth 5 is "
+rect.area());
}
}
```

Output:

```
D:\java>java djsce
Aksh Nishar 60004220123
Area of Square of side 6 is 36.0
Area of Rectangle of length 4 and breadth 5 is 20.0
```

Create Rectangle and Cube class that encapsulates the propertiesof a rectangle and cube i.e. Rectangle has default and parameterized constructor and area() method. Cube has default and parameterized constructor and volume() method. They share no ancestor other thanObject.
Implement a class Size with size() method. This method accepts a single reference argument z. If z refers to a Rectangle then size(z) returns its area and if z is a reference of Cube, then z returns its voliume. If z refers to an object of any other class, then size(z) returns

-1. Use main method in Size class to call size(z) method.

Theory:

The following program implements the use of abstract class where we declare all the functions and define and use it in another class by extending the abstract class.An abstract class is used if you want to provide a common, implemented functionality among all the implementations of the component. Abstract classes willallow you to partially implement your class.

Therefore, it is also known as data hiding, and as discussed previously , Constructor overloading is when there are multiple method which has a same name as the classor constructors , act different on having different parameters , later in this code we have also implemented the class Size with the method size(), it is a method is used to get the size of the Set or the number of elements present in
the Set. Parameterized Constructor – A constructor is called Parameterized Constructor when it accepts a specific number of parameters. To initialize data members of a class with distinct values. With a parameterized constructor for a class, one must provide initial values as arguments, otherwise, the compiler reports an error.

Code:

```
import java.util.*;
class Rect{
private int l,b;
Rect(int l,int b)
{
this.l=l; this.b=b;
}
int area() { return l*b;
}
}
class Cube{ private int side; Cube(int side)
{
this.side=side;
}
int volume(){
return side*side*side;
}
}
class Size{
public static int size(Object o){ if(o instanceof Rect){
return ((Rect)o).area();
}
else if(o instanceof Cube){ return ((Cube)o).volume();
}
else { return -1;
}
}
}
public class Main8{
public static void main(String[] args)
{
System.out.println("Aksh Nishar 60004220123");
Scanner sc = new Scanner(System.in);
Rect r = new Rect(5,6);
Cube c = new Cube(4);
System.out.println("Area of Rectangle : "+Size.size(r));
System.out.println("Volume of Cube : "+Size.size(c));
System.out.println("Other objects : "+Size.size(sc));
}
}
```

Output:

```
D:\java>javac djsce.java

D:\java>java djsce
Aksh Nishar 60004220123
Area of Rectangle : 30
Volume of Cube : 64
Other objects : -1
```

CONCLUSION: Thus we have implemented Constructors and the overloading concept in java.

Name – Aksh Dilip Nishar **SAP ID - 60004220123**

Experiment No - 9

AIM: To implement Abstract classes

Write a *abstract class* program to calculate area of circle, rectangle andtriangle

Theory:

In this below given program we have implemented concepts like data encapsulation , constructor overloading. Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class.
Abstract class called Shape has three subclasses say Triangle,Rectangle,Circle.Method area() in the abstract class and override this area() in these three subclasses to calculate for specific object i.e.area() of Triangle subclass should calculate area of triangle etc. Same for Rectangle and Circle.
An abstract class is like a blueprint/format about the minimum required functions.A method which is declared as abstract and does not have implementation is known as an abstract method.

Code:

```
import java.lang.Math;
 abstract class Shape
{
abstract void area(); double area;
}

class Triangle extends Shape
{
double b=50,h=15; void area()
{
area = (b*h)/2;
System.out.println("area of Triangle -->"+area);
}
}

class Rectangle extends Shape
{
double w=70,h=20; void area()
{
area = w*h;
System.out.println("area of Rectangle -->"+area);
```

```
}
}

class Circle extends Shape
{
double r=5; void area()
{
area = Math.PI * r * r; System.out.println("area of Circle -->"+area);
}
}

class Main9
{
public static void main(String [] args)
{
System.out.println("Aksh Nishar 60004220123");
Triangle t= new Triangle();
Rectangle r =new Rectangle();
Circle c =new Circle();

t.area();
r.area();
c.area();
}
}
```

Output:

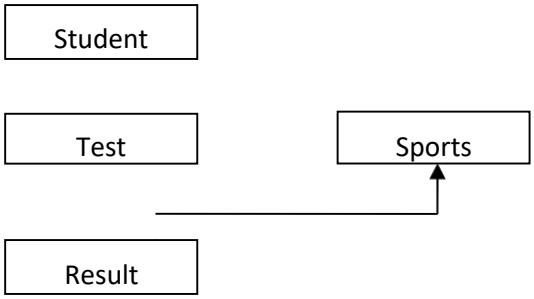
```
Aksh Nishar 60004220123
area of Triangle -->375.0
area of Rectangle -->1400.0
area of Circle -->78.53981633974483
PS D:\java>
```

CONCLUSION: Thus, we have implemented abstract classes in java.

Experiment No - 10

AIM: To implement Inheritance, interfaces and methodOverriding

WAP to implement three classes namely Student, Test and Result. Student class has member as rollno, Test class has members as sem1_marksand sem2_marks and Result class has member as total. Create an interface named sports that has a member score (). Derive Test class from Student andResult class has multiple inheritances from Test and Sports. Total is formula based on sem1_marks, sem2_mark and score.



Theory:

In this program we created and interface named sports which consists of score function and created 3 classes namely student, text by extending the student class and Result by extending student class and implementing the interface sports. Lastly,we created class multiple and executed all the functions.

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. It is an important part of OOPs (Object Oriented programming system). An interface is a reference type, similar to a class.that can containonly constants, method signatures, default methods, static methods, and nested types.

Code:

```
import java.util.Scanner;
interface Sports{
```

```
 int score=100;
 void member_score();
}
class Student{
 int roll_no;
 void read(int n){
 roll_no = n;
 }
 void display(){
 System.out.println(roll_no);
 }
}
class Test extends  Student{
 int sem1_marks,sem2_marks;
 void read1(int n){
 sem1_marks = n;
 }
 void read2(int n){
 sem2_marks = n;
 }
 void display(){
 System.out.println(sem1_marks+sem1_marks);
 }
}
class Result extends Test implements Sports{
 public void member_score(){
 int total;
 total = sem1_marks+sem2_marks+score;
 System.out.println("The total score is "+total);
 }
}
public class Main10 {
 public static void main(String args[]){
 System.out.println("Aksh Nishar 60004220123");
 Scanner s = new Scanner(System.in);
 Result r = new Result();
```

```
 System.out.println("Enter roll no.");
 int roll = s.nextInt();

 System.out.println("Enter sem1.");
 int sem1 = s.nextInt();
 System.out.println("Enter sem2.");
 int sem2 = s.nextInt();
 r.read(roll);
 r.read1(sem1);
 r.read2(sem2);
 r.member_score();
 }
}
```

Output:

```
Aksh Nishar 60004220123
Enter roll no.
259
Enter sem1.
79
Enter sem2.
69
The total score is 248
PS D:\java>
```

CONCLUSION: Thus, we have implemented Inheritance, interfaces and methodOverriding concept in java.