**NAME : SONAL**

**COURSE : BCA**

**ROLL NO : 30**

**SEM : IV ‘C’**

**DATE : 7/2/2020**

**PROBLEM STATEMENT**: write a java program of method overloading to calculate area of circle, area of square, area of rectangle.

**OBJECTIVE:** To understand the concept of method overloading in java.

**DESCRIPTION:** **Method Overloading** is a feature that allows a class to have more than one **method** having the same name, if their argument lists are different. It is similar to constructor **overloading in Java**, that allows a class to have more than one constructor having different argument lists.

**SOURCE CODE:**

class Method

{

void area(float r)

{

System.out.println("Area of circle is : "+(3.14\*r\*r));

}

void area(int side)

{

System.out.println("Area of square is : "+(side\*side));

}

void area(int length,int breath)

{

System.out.println("Area of rectangle is : "+(length\*breath));

}

}

class Overloading

{

public static void main(String[] args)

{

Method m=new Method();

m.area(5);

m.area(6);

m.area(4,7);

}

}

**OUTPUT:**

D:\programs\day>javac Overloading.java

D:\programs\day>java Overloading

Area of square is : 25

Area of square is : 36

Area of rectangle is : 28

D:\programs\day>

**NAME : SONAL**

**COURSE : BCA**

**ROLL NO : 30**

**SEM : IV ‘C’**

**DATE : 7/2/2020**

**PROBLEM STATEMENT**: write a java program to implement default constructor.

**OBJECTIVE:** To understand the concept default constructor of in java.

**DESCRIPTION**: A constructor without parameters is known as default constructor. Constructors are mostly used to initialize the instance variables. Specifically, using default constructors the instance variables will be initialized with fixed values for all objects**.**

**SOURCE CODE:**

class Constructor

{

int roll;

String name;

Constructor()

{

roll=20;

name="hello world";

}

void show()

{

System.out.println("user roll number is :"+roll);

System.out.println("user name is:"+name);

}

public static void main(String[] args)

{

Constructor con=new Constructor();

con.show();

}

}

**OUTPUT:**

D:\programs\day>javac Constructor.java

D:\programs\day>java Constructor

user roll number is :30

user name is:hello java

D:\programs\day>

**NAME : SONAL**

**COURSE : BCA**

**ROLL NO : 30**

**SEM : IV ‘C’**

**DATE : 31/1/2020**

**PROBLEM STATEMENT** : write a java program to find factorial of a number .

**OBJECTIVE:** to understand the concept of command line arguments in java.

**DESCRIPTION:** The **factorial** of a **number** is the product of **all** the integers from 1 to that **number**. For example, the **factorial** of 6 (denoted as 6!) is 1\*2\*3\*4\*5\*6 = 720. **Factorial** is not defined for negative **numbers** and the **factorial** of zero is one, 0!

**SOURCE CODE:**

class Factorial

{

public static void main(String args[])

{

int n,fact=1;

n=Integer.parseInt(args [0]);

while(n>0)

{

fact=fact\*n;

n--;

}

System.out.println("factorial of a number is:"+fact);

}

}

**OUTPUT:**

C:\Users\STUDENT>D:

D:\>cd programs

D:\programs>javac Factorial.java

D:\programs>java Factorial 6

factorial of a number is:720

D:\programs>

**NAME : SONAL**

**COURSE : BCA**

**ROLL NO : 30**

**SEM : IV ‘C’**

**DATE : 7/2/2020**

**PROBLEM STATEMENT**: write a java program to implement Constructor Overloading.

**OBJECTIVE:** To understand the concept of constructor Overloading in java.

**DESCRIPTION**: **Constructor overloading** is a technique in Java in which a class can have any number of **constructors** that differ in parameter list. The compiler differentiates these **constructors** by taking into account the number of parameters in the list and their type.

**SOURCE CODE:**

class Constructor\_Overloading

{

int x,y;

Constructor\_Overloading()

{

x=45;

y=60;

System.out.println("default constructor is called");

}

Constructor\_Overloading(int x,int y)

{

this.x=x;

this.y=y;

System.out.println("parameterized constructor is called");

}

Constructor\_Overloading(Constructor\_Overloading r)

{

x=r.x;

y=r.y;

System.out.println("copy constructor is called");

}

void sum()

{

int z=x+y;

System.out.println("sum="+z);

}

}

class Over

{

public static void main(String[] args)

{

Constructor\_Overloading ob1;

ob1=new Constructor\_Overloading(6,2);

Constructor\_Overloading ob2=new Constructor\_Overloading();

Constructor\_Overloading ob3=new Constructor\_Overloading(ob2);

ob1.sum();

ob2.sum();

ob3.sum();

}

}

**OUTPUT:**

C:\Users\windows\Desktop\java programs>javac Over.java

C:\Users\windows\Desktop\java programs>java Over

parameterized constructor is called

default constructor is called

copy constructor is called

sum=8

sum=105

sum=105

C:\Users\windows\Desktop\java programs>

**NAME : SONAL**

**COURSE : BCA**

**ROLL NO : 30**

**SEM : IV ‘C’**

**DATE : 7/2/2020**

**PROBLEM STATEMENT**: write a java program to implement **Parameterized constructor**.

**OBJECTIVE:** To understand the concept of **Parameterized constructor** in java.

**DESCRIPTION**:  **Parameterized constructor**, A **constructor** having a specific number of parameters(arguments) is called a **parameterized constructor**. The **parameterized constructor** is used to provide different values to the objects, you can also provide the same values.

**SOURCE CODE:**

class Student

{

int id;

String name;

Student(int i, String n)

{

id=i;

name=n;

}

void display()

{

System.out.println(id+" "+name);

}

}

class Parameter

{

public static void main(String[] args)

{

Student st1= new Student(350,"computer");

Student st2= new Student(780,"science ");

st1.display();

st2.display();

}

}

**OUTPUT:**

C:\Users\windows\Desktop\java programs>javac Parameter.java

C:\Users\windows\Desktop\java programs>java Parameter

350 computer

780 science

C:\Users\windows\Desktop\java programs>

**NAME : SONAL**

**COURSE : BCA**

**ROLL NO : 30**

**SEM : IV ‘C’**

**DATE : 7/2/2020**

**PROBLEM STATEMENT**: write a java program to implement **Copy Constructor**.

**OBJECTIVE:** To understand the concept of **Copy Constructor** in java.

**DESCRIPTION:** **Copy Constructor in java** class is a special type of **constructor** that takes same class as argument. **Copy constructor** is used to provide a **copy** of the specified object.

**SOURCE CODE:**

class Student

{

int id;

int roll\_no;

String name;

Student(int i,int r,String n)

{

id=i;

roll\_no=r;

name=n;

}

Student( Student st)

{

id=st.id;

roll\_no=st.roll\_no;

name=st.name;

}

void display()

{

System.out.println(id+" "+roll\_no+" "+name);

}

}

class Copy

{

public static void main(String[] args)

{

Student st1=new Student(18215445,78,"World");

Student st2=new Student(st1);

st1.display();

st2.display();

}

}

**OUTPUT**

C:\Users\windows\Desktop\java programs>javac Copy.java

C:\Users\windows\Desktop\java programs>java Copy

18215445 78 World

18215445 78 World

C:\Users\windows\Desktop\java programs>