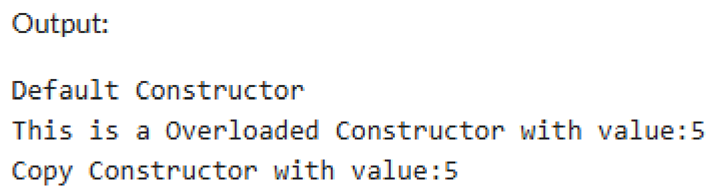
Practical 1A:

```
class Myclass{
  private int a;
  public Myclass(){
    System.out.println("Default Constructor");
  }
  public Myclass(int val){
    a=val;
    System.out.println("This is a Overloaded Constructor with value:"+a);
  }
  public Myclass(Myclass other){
    a=other.a;
    System.out.println("Copy Constructor with value:"+a);
  }
}
public class MainClass{
  public static void main(String args[]){
    Myclass obj1=new Myclass();
    Myclass obj2=new Myclass(5);
    Myclass obj3=new Myclass(obj2);
  }
}
```

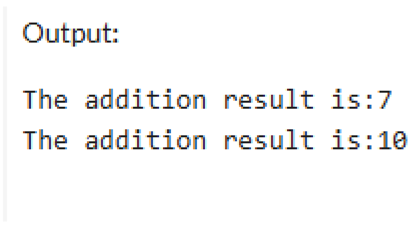


Output:

```
Default Constructor
This is a Overloaded Constructor with value:5
Copy Constructor with value:5
```

Practical 1B:

```
class MethodOver{

  public void Add(int val1,int val2){
    int a=val1+val2;
    System.out.println("The addition result is:"+a);
  }
  public void Add(int val1,int val2,int val3){
    int a=val1+val2+val3;
    System.out.println("The addition result is:"+a);
  }
}
public class MainClass{
  public static void main(String args[]){
    MethodOver obj1=new MethodOver();
    obj1.Add(2,5);
    MethodOver obj2=new MethodOver();
    obj2.Add(2,5,3);
  }
}
```

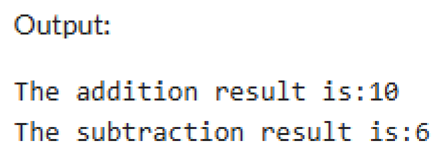


Output:

```
The addition result is:7
The addition result is:10
```

Practical 1C:

```
class StaticMethods{

  public static void add(int val1,int val2){
    int a=val1+val2;
    System.out.println("The addition result is:"+a);
  }
  public static void sub(int val1,int val2){
    int a=val1-val2;
    System.out.println("The subtraction result is:"+a);
  }
}
public class MainClass{
  public static void main(String args[]){
StaticMethods.add(3,7);
StaticMethods.sub(10,4);
  }
}
```

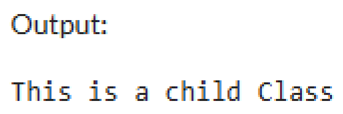


Output:

```
The addition result is:10
The subtraction result is:6
```

Practical 2A:

```
class parent{
  void message(){
    System.out.println("Parent Class");
  }
}
class child extends parent{
  void message(){
    System.out.println("This is a child Class");
  }
}
public class Inherit{
  public static void main(String args[]){
child c=new child();
c.message();
  }
}
```



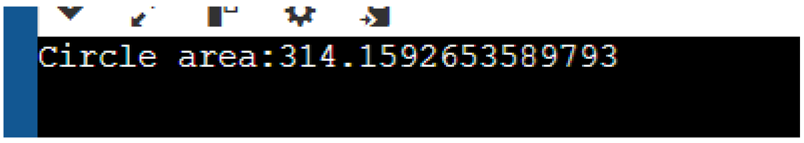
Output:

```
This is a child Class
```

Practical 2B:

```
abstract class shape{
    public abstract double area();
}
class circle extends shape{
    private double radius;
    public circle(double radius){
        this.radius=radius;
    }
    @Override
    public double area(){
        return Math.PI*radius*radius;
    }
}
public class Main
{
            public static void main(String[] args) {
                circle obj=new circle(10.0);
                System.out.println("Circle area:"+obj.area());
        }
}
```

Output:

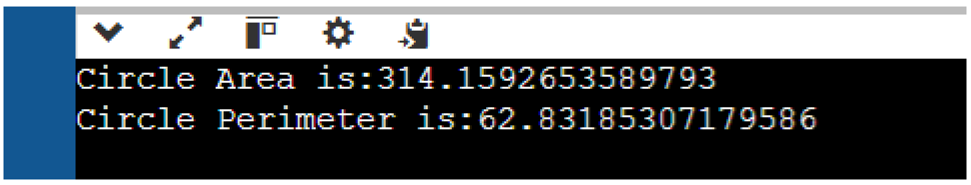


```
Circle area:314.1592653589793
```

Practical 2C:

```
interface shape{
    double area();
    double perimeter();
}
class circle implements shape{
    private double radius;
    public circle(double radius){
        this.radius=radius;
    }
    @Override
    public double area(){
        return Math.PI*radius*radius;
    }
    @Override
    public double perimeter(){
        return 2*Math.PI*radius;
    }
}
public class Main{
    public static void main(String args[]){
        circle obj=new circle(10.0);
        System.out.println("Circle Area is:"+obj.area());
         System.out.println("Circle Perimeter is:"+obj.perimeter());
    }
}
```

Output:

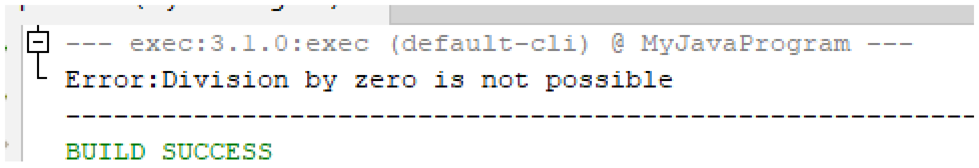


```
Circle Area is:314.1592653589793
Circle Perimeter is:62.83185307179586
```

Practical 3A:

```
class err{
   public int div(int val1,int val2){
      return val1/val2;
   }
}
public class MyJavaProgram {

   public static void main(String[] args) {
      try{
      err obj=new err();
      System.out.println(obj.div(10,0));
   }
      catch(ArithmeticException e){
         System.out.println("Error:Division by zero is not possible");
      }
   }
}
```

Output:

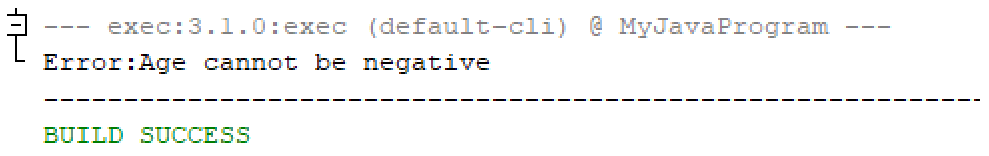


```
--- exec:3.1.0:exec (default-cli) @ MyJavaProgram ---
Error:Division by zero is not possible
------------------------------------------------------------------------
BUILD SUCCESS
```

Practical 3B:

```
class CustomExc extends Exception{
   public CustomExc(String message){
      super(message);
   }
}
public class MyJavaProgram{
   public static void main(String[] args){
      try{
         int age=-9;
         if(age<0){
            throw new CustomExc("Age cannot be negative");
         }
         System.out.println("Age:"+age);
      }
      catch(CustomExc e){
         System.err.println("Error:"+e.getMessage());
      }
   }
}
```

Output:

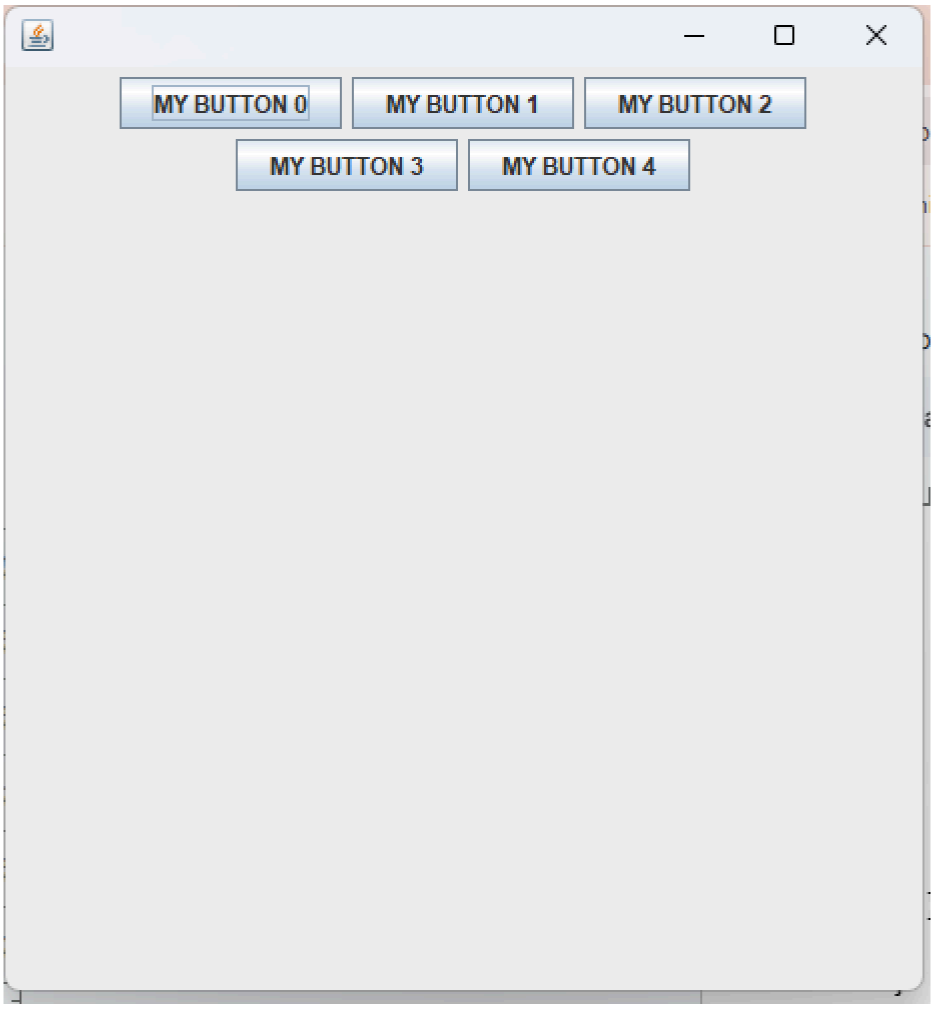


```
--- exec:3.1.0:exec (default-cli) @ MyJavaProgram ---
Error:Age cannot be negative
------------------------------------------------------------
BUILD SUCCESS
```

Practical 7a:

```
package javaapplication1;

import java.awt.FlowLayout;
import javax.swing.JButton;
import javax.swing.JFrame;

public class JavaApplication1 {


    public static void main(String[] args) {
        JFrame frame=new JFrame();

        FlowLayout layout =new FlowLayout();
        frame.setLayout(layout);

        for(int i=0;i<5;i++){
            JButton button=new JButton("MY BUTTON "+i);
            frame.add(button);
        }
        frame.setDefaultCloseOperation(3);
        frame.setSize(500,500);
        frame.setVisible(true);

    }

}
```

Output:

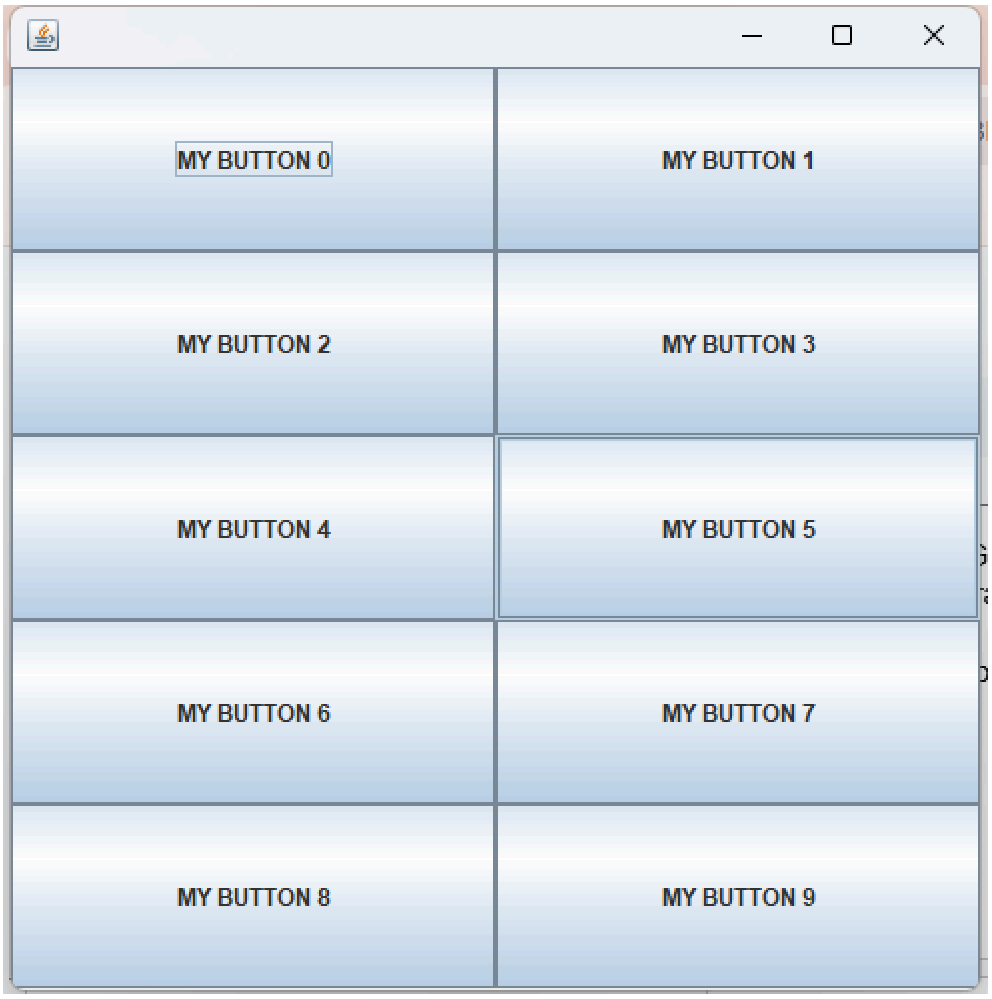


Practical 7b:

```
package javaapplication1;
import java.awt.GridLayout;
import javax.swing.JButton;
import javax.swing.JFrame;

public class JavaApplication1 {


   public static void main(String[] args) {
      JFrame frame=new JFrame();

      GridLayout layout =new GridLayout(5,2);
      frame.setLayout(layout);

      for(int i=0;i<10;i++){
         JButton button=new JButton("MY BUTTON "+i);
```

```
            frame.add(button);
        }
        frame.setDefaultCloseOperation(3);
        frame.setSize(500,500);
        frame.setVisible(true);

    }

}
```

Output:

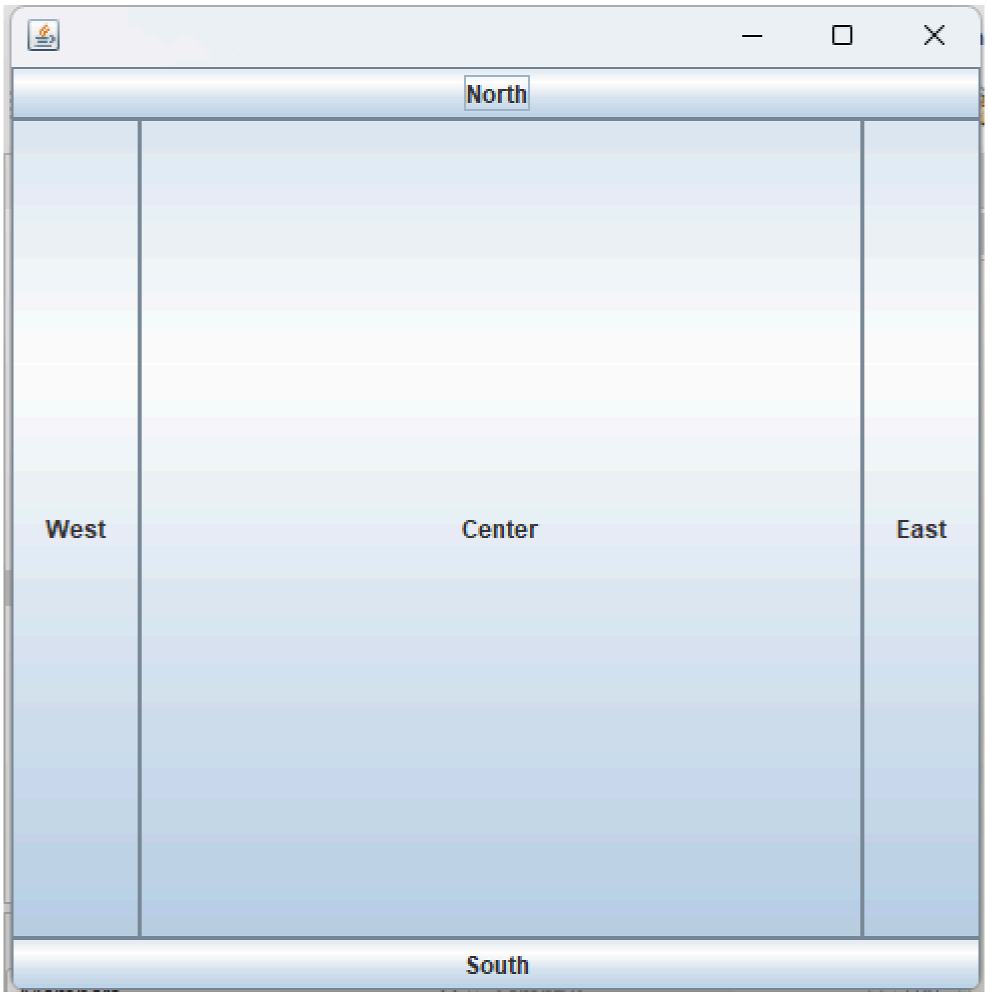


Practical 7c:

```
package javaapplication1;

import javax.swing.JFrame;
import java.awt.BorderLayout;
import javax.swing.JButton;
```

```
public class BorderLayoutExample {
    public static void main(String[] args){
        JFrame frame=new JFrame();
        BorderLayout layout=new BorderLayout();
        frame.setLayout(layout);
        JButton north=new JButton("North");
        JButton east=new JButton("East");
        JButton west=new JButton("West");
        JButton south=new JButton("South");
        JButton center=new JButton("Center");
        frame.add(north,BorderLayout.NORTH);
        frame.add(east,BorderLayout.EAST);
        frame.add(west,BorderLayout.WEST);
        frame.add(south,BorderLayout.SOUTH);
        frame.add(center,BorderLayout.CENTER);
        frame.setSize(500,500);
        frame.setDefaultCloseOperation(3);
        frame.setVisible(true);
    }

}
```

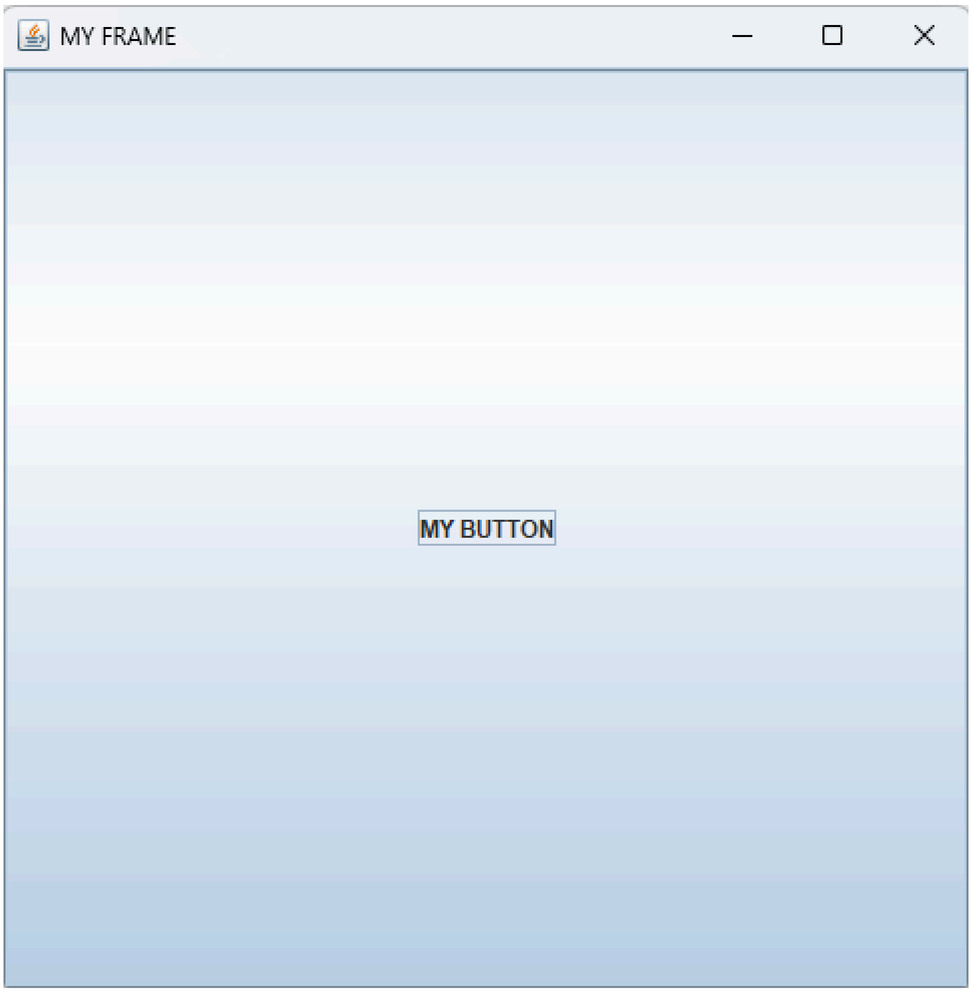
Output:

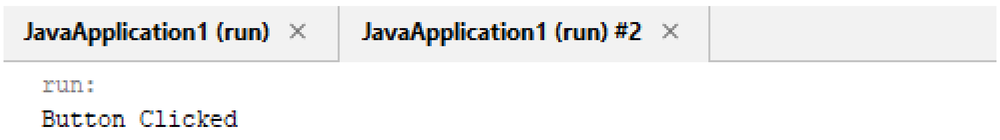


Practical 8a & also for 10:

```
package javaapplication1;


import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;


public class EventHandlerExample {
   public static void main(String[] args) {
      JFrame frame=new JFrame("MY FRAME");

      JButton button=new JButton("MY BUTTON");
      //Anonymous inner class
      button.addActionListener(new ActionListener(){
         public void actionPerformed(ActionEvent e){
            System.out.println("Button Clicked");
         }
      });

      frame.add(button);

      frame.setSize(500,500);
      frame.setDefaultCloseOperation(3);
      frame.setVisible(true);

   }

}
```

Output:



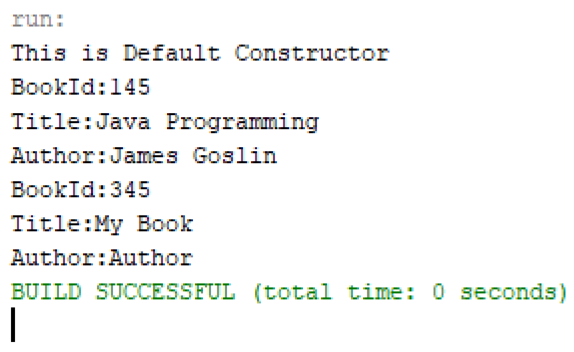


```
run:
Button Clicked
```

PRACTICAL no.1a:(Revision)

```
public class Book {
    int bookid;
    String title;
    String author;
    public void Display(){
        System.out.println("BookId:"+bookid);
        System.out.println("Title:"+title);
        System.out.println("Author:"+author);
    }
    public Book(){
        System.out.println("This is Default Constructor");
    }
    public Book(int ID,String TITLE,String AUTHOR){
        this.bookid=ID;
        this.title=TITLE;
        this.author=AUTHOR;
    }
    public static void main(String[] args) {
        Book obj1=new Book();
        obj1.bookid=145;
        obj1.title="Java Programming";
        obj1.author="James Goslin";
        obj1.Display();

        Book obj2=new Book(345,"My Book","Author");
        obj2.Display();
    }

}
```

Output:

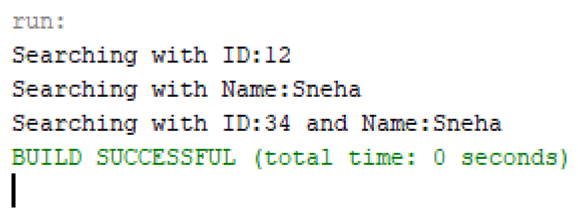


```
run:
This is Default Constructor
BookId:145
Title:Java Programming
Author:James Goslin
BookId:345
Title:My Book
Author:Author
BUILD SUCCESSFUL (total time: 0 seconds)
```

PRACTICAL no.1b:(Revision)

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */

/**
 *
 * @author Dell
 */
public class MethodOverloading {
    public void search(int a){
        System.out.println("Searching with ID:"+a);
    }
    public void search(String name){
        System.out.println("Searching with Name:"+name);
    }
    public void search(int b,String s){
        System.out.println("Searching with ID:"+b+" and Name:"+s);
    }
    public static void main(String[] args) {
        MethodOverloading obj1=new MethodOverloading();
        obj1.search(12);
        obj1.search("Sneha");
        obj1.search(34,"Sneha");

    }

}
```

Output:



```
run:
Searching with ID:12
Searching with Name:Sneha
Searching with ID:34 and Name:Sneha
BUILD SUCCESSFUL (total time: 0 seconds)
```