Shreyas Sawant
D7A - 55

AIM: To implement interfaces in Java

THEORY:

- Interface:

Using the keyword interface, you can fully abstract a class' interface from its implementation. That is, using interface, you can specify what a class must do, but not how it does it. Interfaces are similar to class in syntax, but they lack instance variables and their methods are declared without any body.

Once it is defined, any number of classes can implement an interface. To implement an interface, a class must create the complete set of methods defined by the interface. Also, one class can implement many interfaces.

Interfaces are designed to support dynamic method resolution at run time. Normally, in order for a method to be called from one class to another both class need to be present at compile time so the Java compiler can check to ensure that the method signatures are environment compatible. However interfaces are designed to avoid this problem, since they are in a different hierarchy from classes, it is possible for classes that are unrelated in terms of class hierarchy to implement the same interface.

Syntax:

```
access interface name:{
    final type varname1= value;
    return type methodname ( parameter list);
}
```

For eg:

```
interface Seen {
  void  to look();
   void  peep(int a);
}

class A   implements   seen
{      void tolook()
    { System.out.println("5");
     }

      void peep (int W
    { System.out.println (k);
     }
}
class Test
{   public void ~~ma~~ static main(string args[])
     {         A  a= new A();
               a.tolook();
               a. to peep(6);
         }
}
```

Output

```
5
6
```

- ## Partial Implementations

If a class includes an interface but does not fully implement the methods defined by that interface, then that class must be abstract and this type of implementation is known as partial implementation.

For eg:

```
abstract class Try implements Seen
{  int a,b;
   void tolook();
   { System.out.println(a+" "+b);
   }
}
```

- ## Extending and Implementing

```
interface A
{    void show();
}

class B
{  int i;
   void bear();
    { System.out.println("The value is: ");
    }
}

class C extends C implements A
{   void show()
    { System.out.println(*i);
    }
}
```

- Extending Interfaces.

```
 interface A
{  void meth1();
   void meth2();
 }
interface B extends A
{  void meth3();
 }
class  Light implements B
{    public void meth1()
    { System.out.println (" Method 1"); }
     public void meth2()
     { System.out.println (" Method 2"); }
      public void meth3()
      { System.out.println (" Method 3"); }
 }
class Test
{  public static void main(String args[])
    {    Light ob = new Light();
         ob.meth1();
         ob.meth2();
         ob.meth3();
    }
 }
```

CONCLUSION

- Errors encountered

1) ~~I~~ Wrong method call in MatInterface main method.

```
ob1. calcmath();
```

Solution The calcmath() method does not exist whereas required method is ob1-calcmat();

2) ArrayIndexOutOfBounds in MatInterface main method

```
for(int i=0; i<4; i++)
{ - - - }
```

S.

Solution The size of declared array is (3,3). Therefore correct syntax

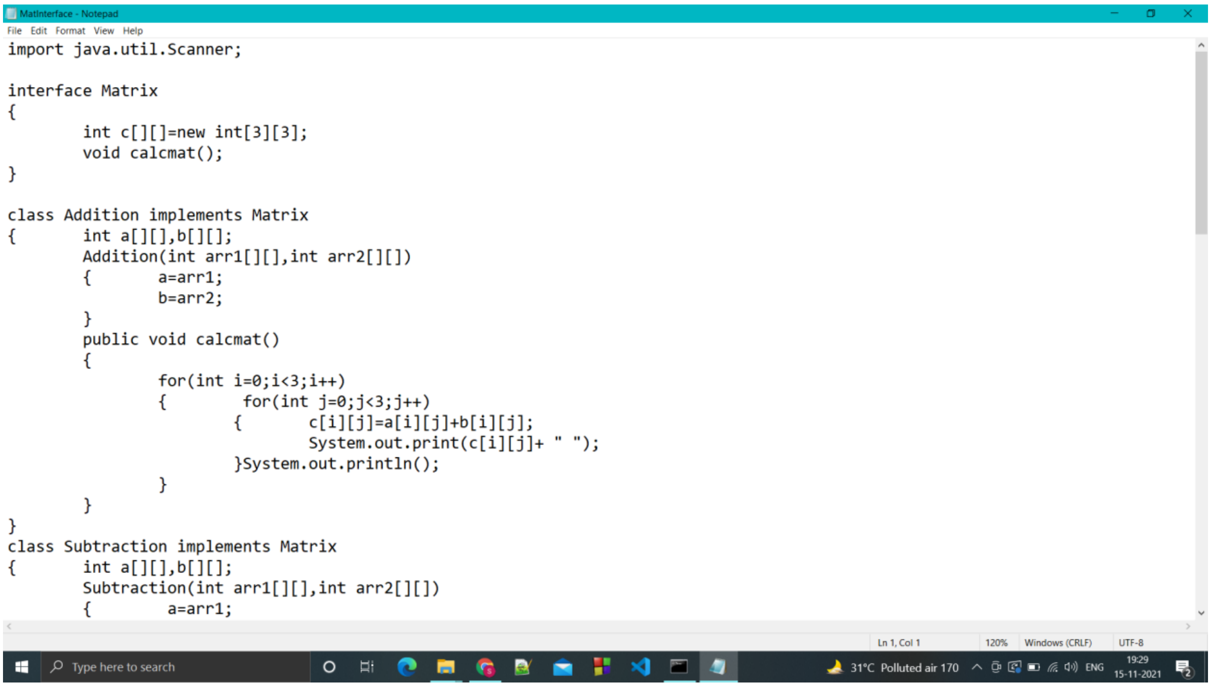
```
for (int i=0; i< 3; i++)
{ - - - - - }
```

LAB 7: INTERFACES IN JAVA

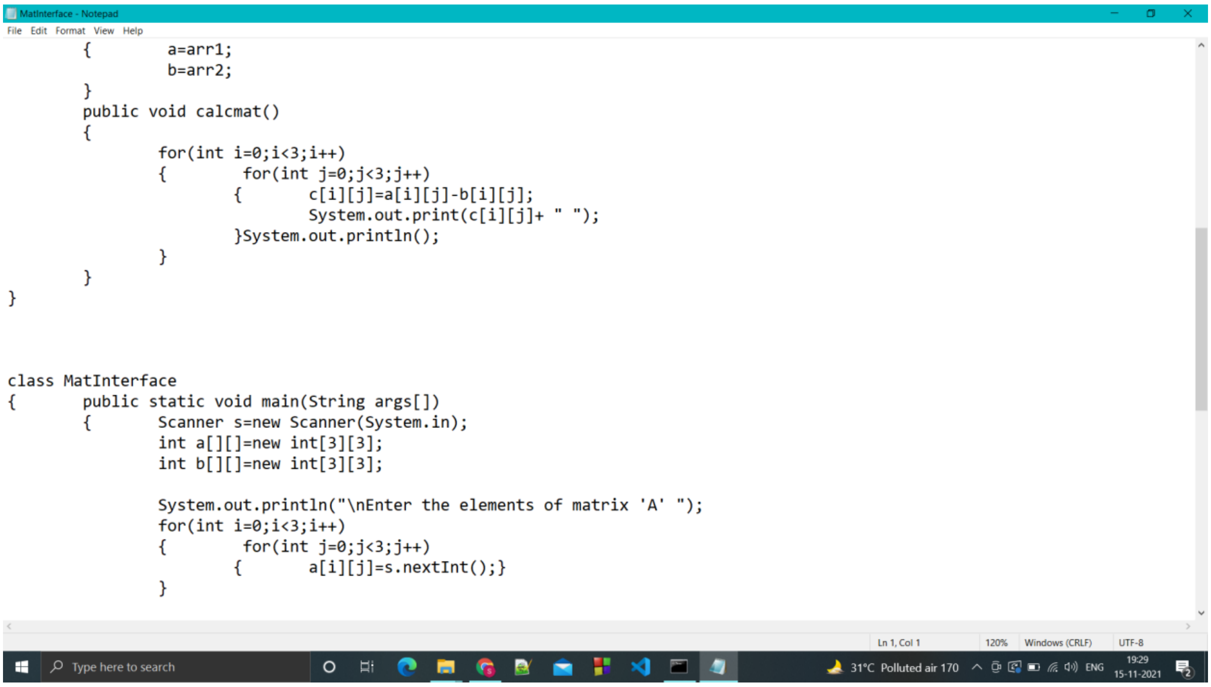
Name: Shreyas Sawant Div: D7A Roll No.: 55

Q.1 Write a program to define the interface called Matrix. Take the maximum number of rows and columns to be 3 and perform matrix addition and subtraction while implementing the given interfaces.

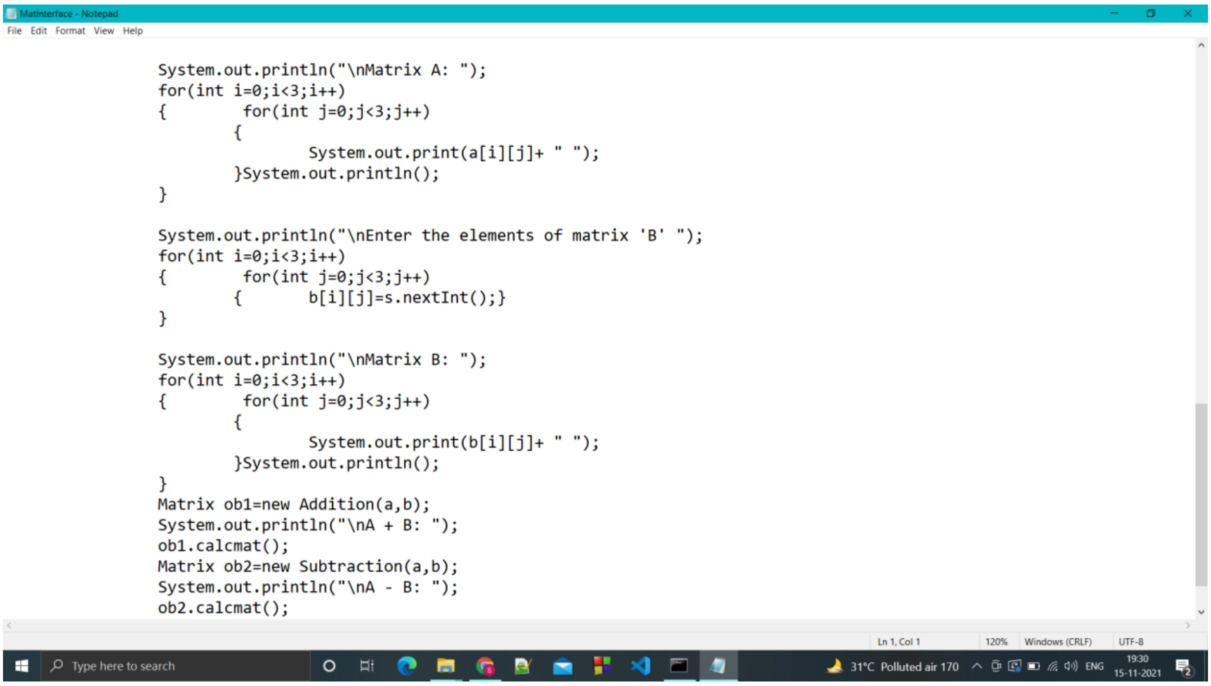
CODE:



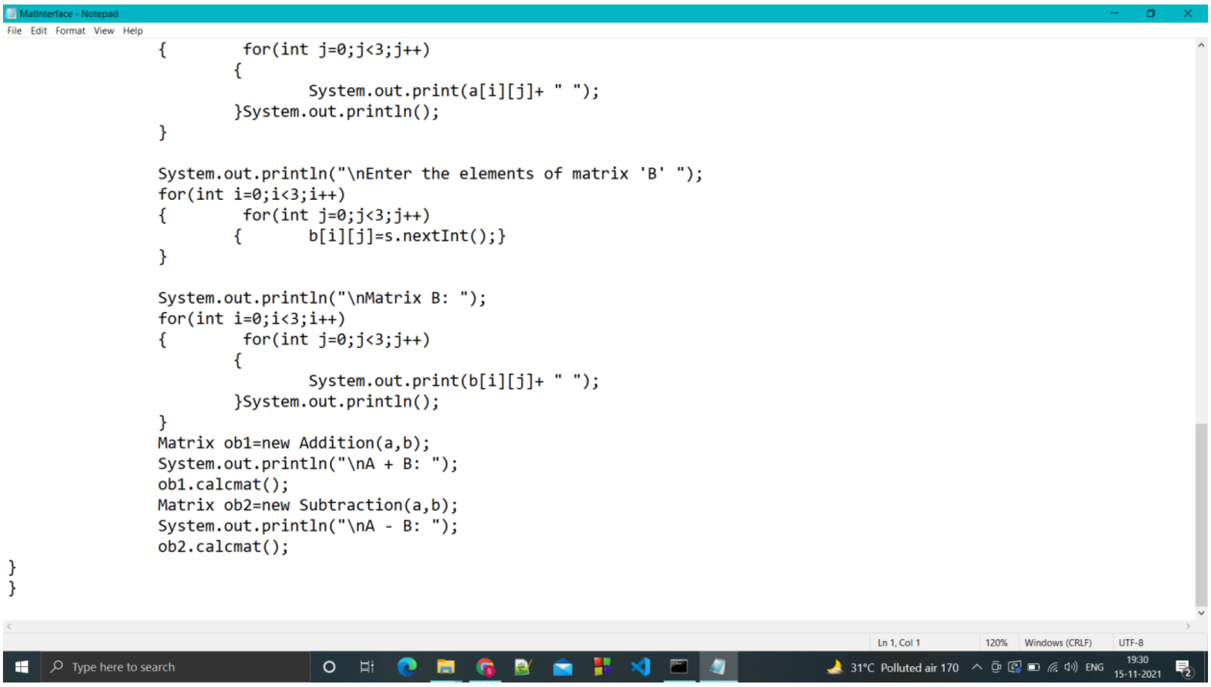
```
import java.util.Scanner;

interface Matrix
{
        int c[][]=new int[3][3];
        void calcmat();
}

class Addition implements Matrix
{       int a[][],b[][];
        Addition(int arr1[][],int arr2[][])
        {       a=arr1;
                b=arr2;
        }
        public void calcmat()
        {
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {       c[i][j]=a[i][j]+b[i][j];
                                System.out.print(c[i][j]+ " ");
                        }System.out.println();
                }
        }
}
class Subtraction implements Matrix
{       int a[][],b[][];
        Subtraction(int arr1[][],int arr2[][])
        {        a=arr1;
```



```
        {        a=arr1;
                 b=arr2;
        }
        public void calcmat()
        {
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {       c[i][j]=a[i][j]-b[i][j];
                                System.out.print(c[i][j]+ " ");
                        }System.out.println();
                }
        }
}



class MatInterface
{       public static void main(String args[])
        {       Scanner s=new Scanner(System.in);
                int a[][]=new int[3][3];
                int b[][]=new int[3][3];

                System.out.println("\nEnter the elements of matrix 'A' ");
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {       a[i][j]=s.nextInt();}
                }
```

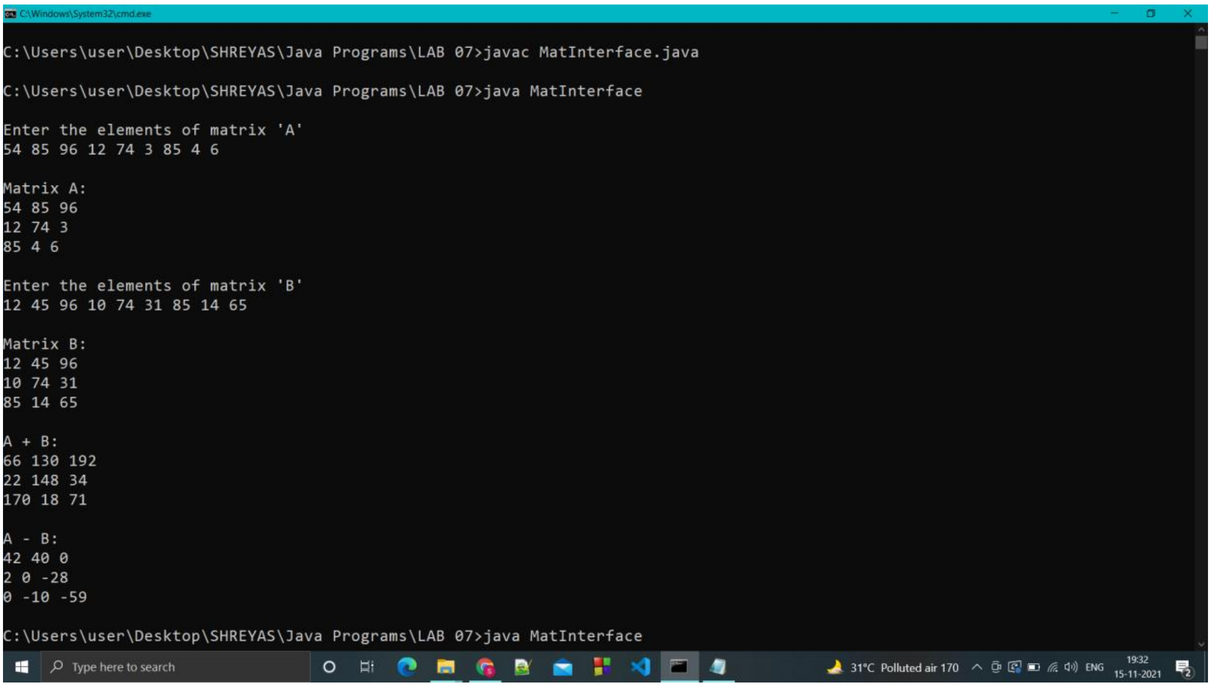


```
                System.out.println("\nMatrix A: ");
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {
                                System.out.print(a[i][j]+ " ");
                        }System.out.println();
                }

                System.out.println("\nEnter the elements of matrix 'B' ");
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {       b[i][j]=s.nextInt();}
                }

                System.out.println("\nMatrix B: ");
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {
                                System.out.print(b[i][j]+ " ");
                        }System.out.println();
                }
                Matrix ob1=new Addition(a,b);
                System.out.println("\nA + B: ");
                ob1.calcmat();
                Matrix ob2=new Subtraction(a,b);
                System.out.println("\nA - B: ");
                ob2.calcmat();
```



```
                {        for(int j=0;j<3;j++)
                        {
                                System.out.print(a[i][j]+ " ");
                        }System.out.println();
                }

                System.out.println("\nEnter the elements of matrix 'B' ");
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {       b[i][j]=s.nextInt();}
                }

                System.out.println("\nMatrix B: ");
                for(int i=0;i<3;i++)
                {        for(int j=0;j<3;j++)
                        {
                                System.out.print(b[i][j]+ " ");
                        }System.out.println();
                }
                Matrix ob1=new Addition(a,b);
                System.out.println("\nA + B: ");
                ob1.calcmat();
                Matrix ob2=new Subtraction(a,b);
                System.out.println("\nA - B: ");
                ob2.calcmat();
}
}
```

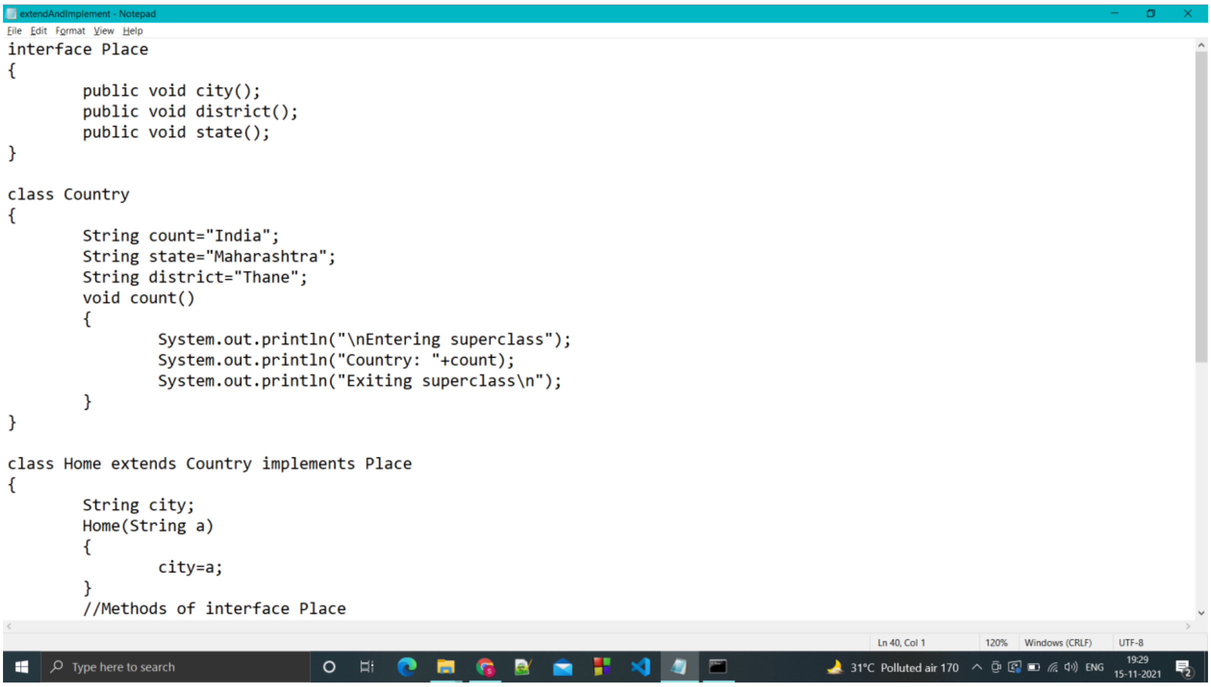
OUTPUT:



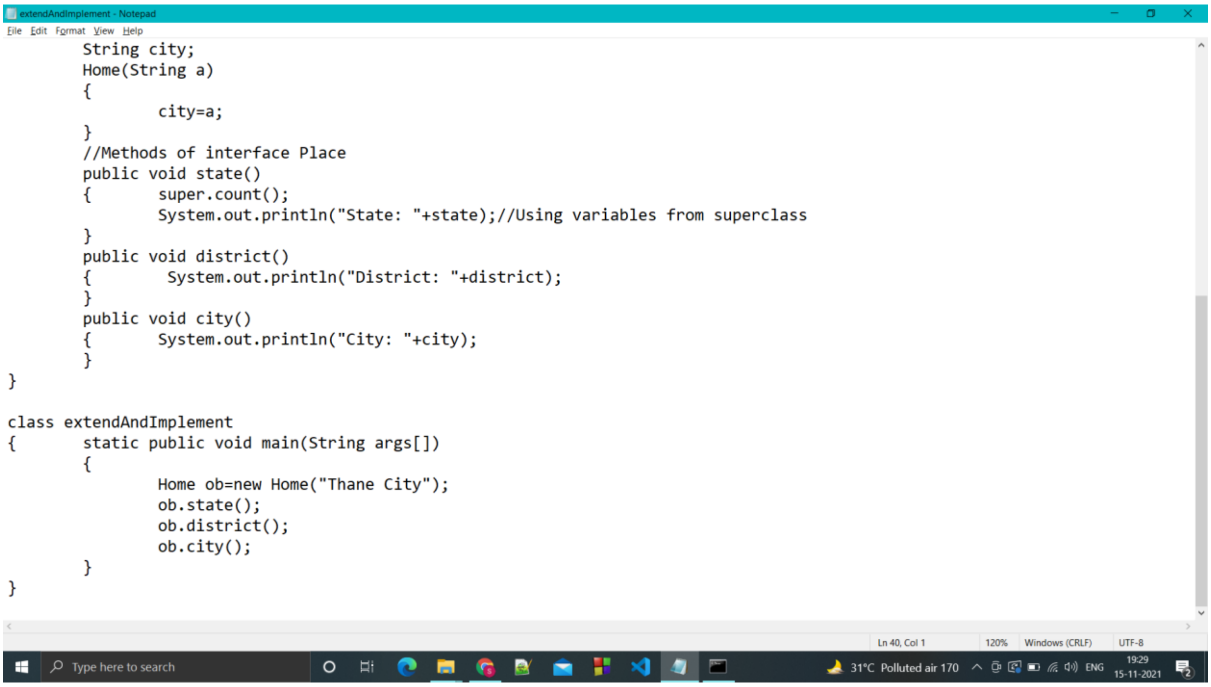
```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>javac MatInterface.java

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>java MatInterface

Enter the elements of matrix 'A'
54 85 96 12 74 3 85 4 6

Matrix A:
54 85 96
12 74 3
85 4 6

Enter the elements of matrix 'B'
12 45 96 10 74 31 85 14 65

Matrix B:
12 45 96
10 74 31
85 14 65

A + B:
66 130 192
22 148 34
170 18 71

A - B:
42 40 0
2 0 -28
0 -10 -59

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>java MatInterface
```

Q.2 Write a program to demonstrate how a class and an interface can be extended and implemented at the same time.

CODE:

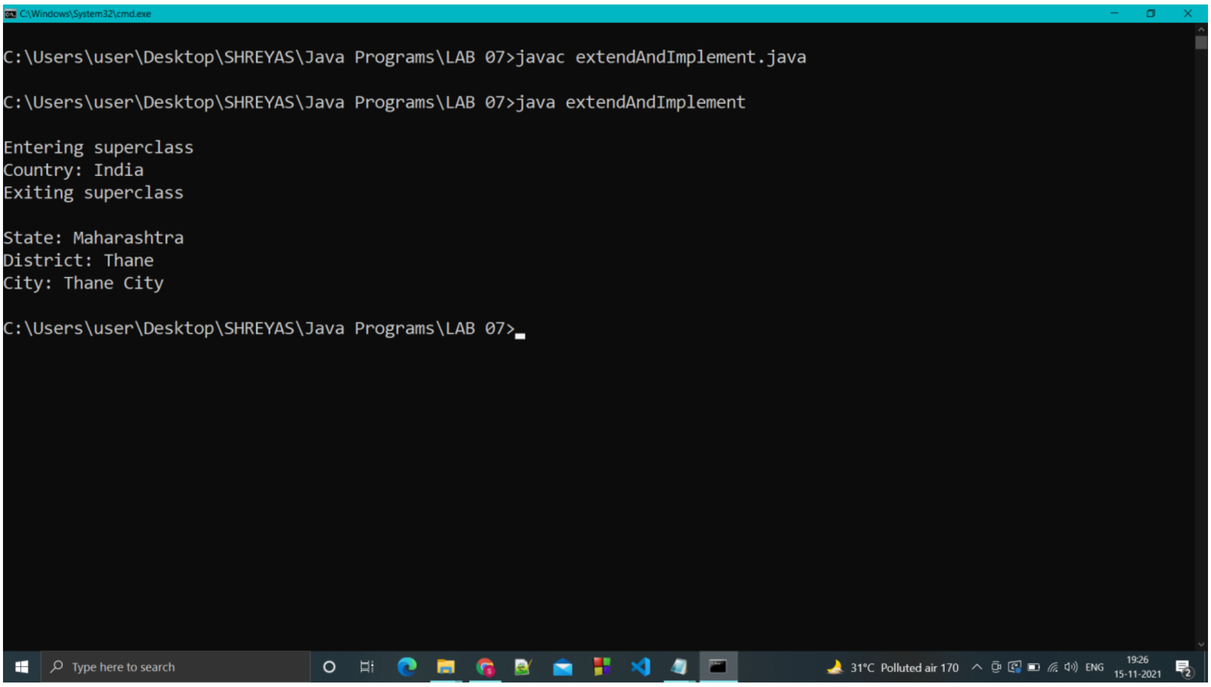


```
interface Place
{
        public void city();
        public void district();
        public void state();
}

class Country
{
        String count="India";
        String state="Maharashtra";
        String district="Thane";
        void count()
        {
                System.out.println("\nEntering superclass");
                System.out.println("Country: "+count);
                System.out.println("Exiting superclass\n");
        }
}

class Home extends Country implements Place
{
        String city;
        Home(String a)
        {
                city=a;
        }
        //Methods of interface Place
```



```
        String city;
        Home(String a)
        {
                city=a;
        }
        //Methods of interface Place
        public void state()
        {       super.count();
                System.out.println("State: "+state);//Using variables from superclass
        }
        public void district()
        {        System.out.println("District: "+district);
        }
        public void city()
        {       System.out.println("City: "+city);
        }
}

class extendAndImplement
{       static public void main(String args[])
        {
                Home ob=new Home("Thane City");
                ob.state();
                ob.district();
                ob.city();
        }
}
```

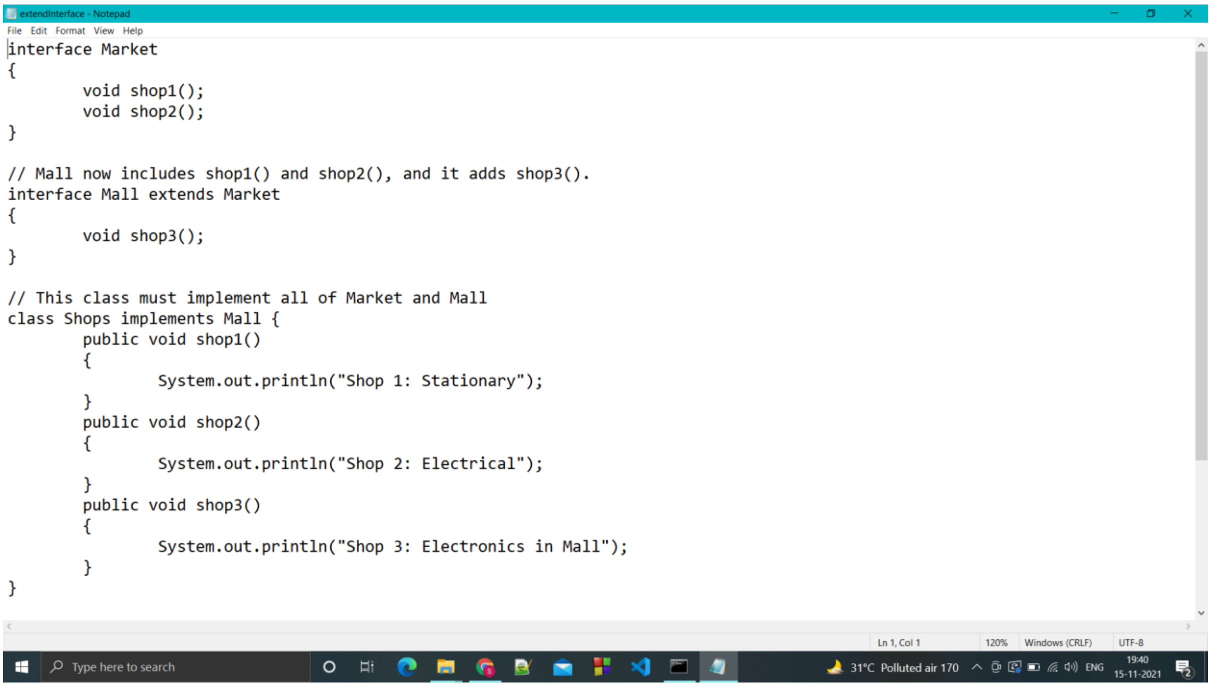
OUTPUT:



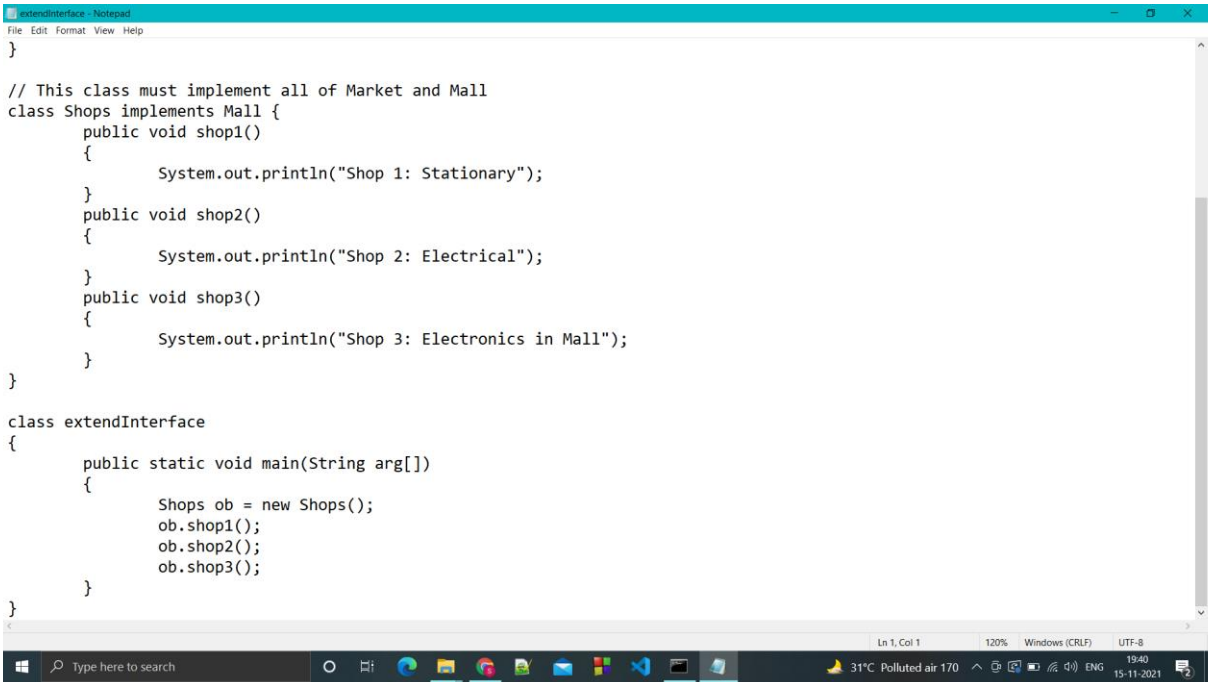
```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>javac extendAndImplement.java

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>java extendAndImplement

Entering superclass
Country: India
Exiting superclass

State: Maharashtra
District: Thane
City: Thane City

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>_
```

Q.3 Write a program to demonstrate how interfaces can be extended.

CODE:



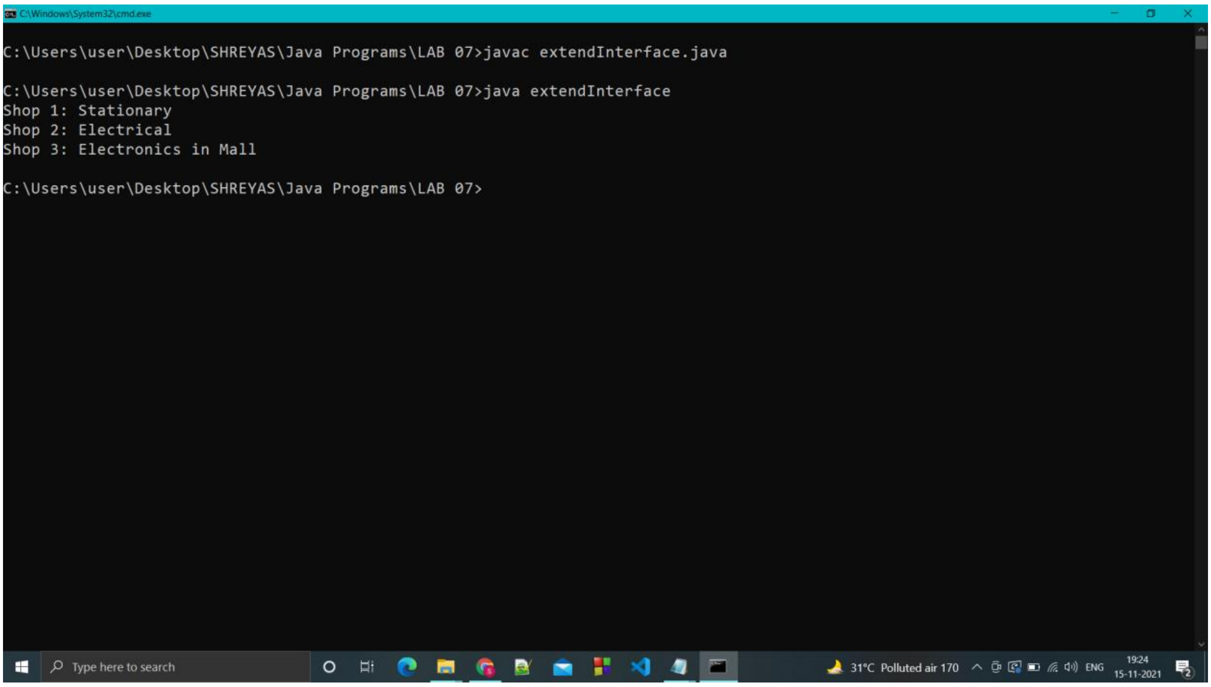
```
interface Market
{
        void shop1();
        void shop2();
}

// Mall now includes shop1() and shop2(), and it adds shop3().
interface Mall extends Market
{
        void shop3();
}

// This class must implement all of Market and Mall
class Shops implements Mall {
        public void shop1()
        {
                System.out.println("Shop 1: Stationary");
        }
        public void shop2()
        {
                System.out.println("Shop 2: Electrical");
        }
        public void shop3()
        {
                System.out.println("Shop 3: Electronics in Mall");
        }
}
```



```
}

// This class must implement all of Market and Mall
class Shops implements Mall {
        public void shop1()
        {
                System.out.println("Shop 1: Stationary");
        }
        public void shop2()
        {
                System.out.println("Shop 2: Electrical");
        }
        public void shop3()
        {
                System.out.println("Shop 3: Electronics in Mall");
        }
}
```

```
class extendInterface
{
        public static void main(String arg[])
        {
                Shops ob = new Shops();
                ob.shop1();
                ob.shop2();
                ob.shop3();
        }
}
```

OUTPUT:



```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>javac extendInterface.java

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>java extendInterface
Shop 1: Stationary
Shop 2: Electrical
Shop 3: Electronics in Mall

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 07>
```