Q.1) Write a program to implement I/O decorator for converting uppercase letters to Lowercase letters.

1. Create Interface:

```
public interface ToLowerDecorator {
	public void lower(String ch);
}
```

2. LowerCase.java

```
package javaprograms;
import java.lang.*;
import java.io.*;
public class LowerCase implements ToLowerDecorator{
	public void lower(String ch)
	{
		ch=ch.toLowerCase();
		System.out.println("Lowercase:"+ch);
	}
}
```

3. Decorator.java

```
package javaprograms;
import java.io.*;
import java.util.Scanner;
public class Decorator {
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		ToLowerDecorator l=new LowerCase();
		//l.lower("HeLLO");
		Scanner sc=new Scanner(System.in);
	 System.out.println("enter character:");
	 String   s=sc.nextLine();
	 System.out.println("entered character:"+s);
	   l.lower(s);
	}
}
```

Output –

```
enter character:
HELLO
entered character:HELLO
Lowercase:hello
```

2) Write a python program to prepare scatter plot for Iris dataset.

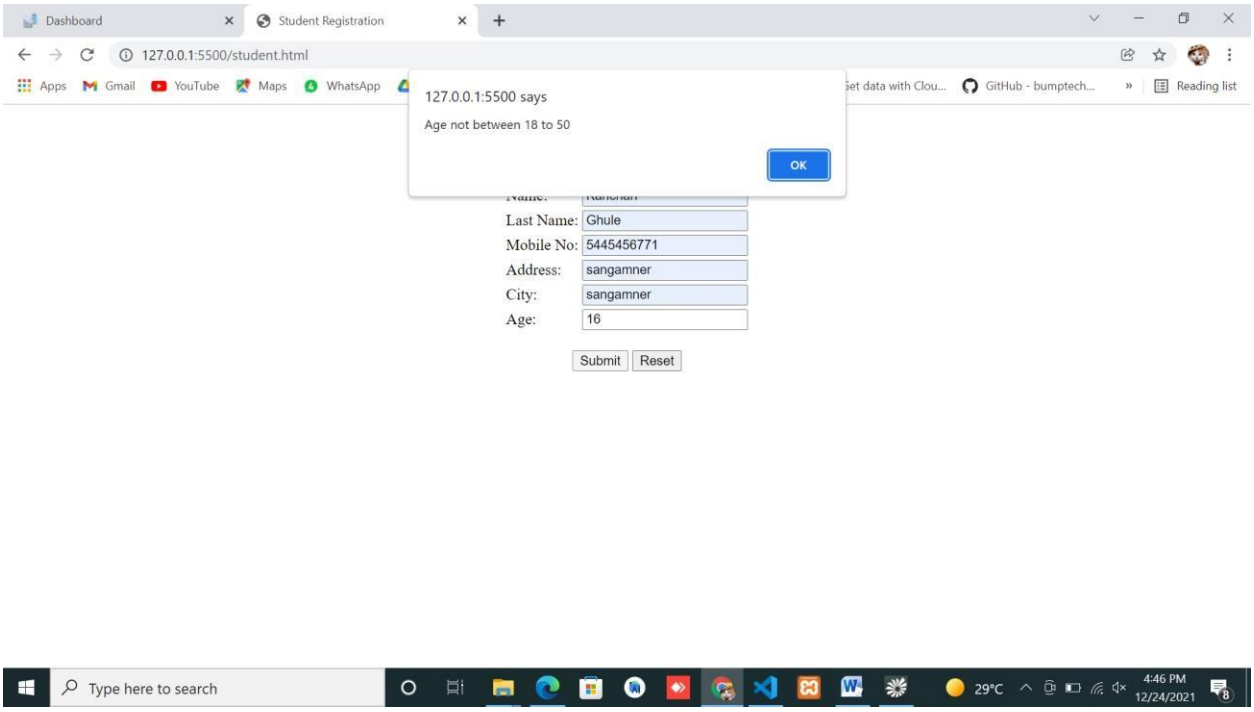
```
import pandas as pd
import matplotlib.pyplot as plt
iris=pd.read_csv("Iris.csv")
iris.plot(kind="scatter",x='SepalLengthCm',y='PetalLengthCm')
```

```
plt.grid()
plt.show()
```

Q.3) Create an HTML form that contain the Student Registration details and write a JavaScript to validate Student first and last name as it should not contain other than alphabets and age should be between 18 to 50.

```
<!DOCTYPE html>
<html>
<head>
   <title>Student Registration</title>
</head>
<body>
   <script type="text/javascript">
      function validation() {
         var name = document.frm.name.value;
         var lname = document.frm.lname.value;
         var mobile = document.frm.mob.value;
         var address = document.frm.add.value;
         var city = document.frm.city.value;
         var age = document.frm.age.value;
         if (name != "" && lname != "" && mobile != "" && address != "" && city != "" && age
!= "") {
            if (!name.match(/^[a-z A-Z]+$/)) {
               alert("Invalid name fields");
            } else if (!lname.match(/^[a-z A-Z]+$/)) {
               alert("Invalid last name fields");
            } else if (!mobile.match(/^\d{10}$/)) {
               alert("Invalid mobile no fields");
            }else if(age>=18 && age<=50) {
               alert("Submit succesfully ..... ");
            } else {
                  alert("Age not between 18 to 50")
            }
         }
         else {
            alert("Invalid fields");
         }
      }
   </script>
   <center>
      <h1>Student Registration Form</h1>
      <form name="frm">
         <table>
            <tr>
               <td>Name: </td>
               <td><input type="text" name="name"></td>
```

```
            </tr>
            <tr>
               <td>Last Name: </td>
               <td><input type="text" name="lname"></td>
            </tr>
            <tr>
               <td>Mobile No: </td>
               <td><input type="text" name="mob"></td>
            </tr>
            <tr>
               <td>Address: </td>
               <td><input type="text" name="add"></td>
            </tr>
            <tr>
               <td>City: </td>
               <td><input type="text" name="city"></td>
            </tr>
            <tr>
               <td>Age: </td>
               <td><input type="text" name="age"></td>
            </tr>
         </table><br>
         <input type="button" name="b1" onclick="validation()" value="Submit">
         <input type="reset" name="b2">
      </form>
</center>
</body>
</html>
```


Dashboard
Student Registration
127.0.0.1:5500/student.html
Apps
Gmail
YouTube
Maps
WhatsApp
GitHub - bumptech...
Reading list
127.0.0.1:5500 says
Age not between 18 to 50
OK
Last Name:
Ghule
Mobile No:
5445456771
Address:
sangamner
City:
sangamner
Age:
16
Submit
Reset
Type here to search
29°C
4:46 PM
12/24/2021

Slip-2)

Q.1) Write a java program to implement singleton pattern for multithreading.

```
package javaprograms;
public class SingletoneTest
{
	private static final int PROCESSOR_COUNT =Runtime.getRuntime().availableProcessors();
	private static final Thread[] THREADS = new Thread[PROCESSOR_COUNT];
	private static int instancesCount = 0;
	private static SingletoneTest instance = null;

	/*** private constructor to prevent Creation of Object fromOutside of the * This class.
	  */
	private SingletoneTest()
	{
	}
	/*** return the instance only if it does not exist */
	public static SingletoneTest getInstance()
	{
		if (instance == null)
		{
			instancesCount++;
			instance = new SingletoneTest();
		}
		return instance;
	}
	/*** reset instancesCount and instance.*/
	private static void reset()
	{
		instancesCount = 0;
		instance = null;
	}
	/*** validate system to run the test*/
	private static void validate()
	{
		if (SingletoneTest.PROCESSOR_COUNT < 2)
		{
			System.out.print("PROCESSOR_COUNT Must be >= 2 to Run the test.");
			System.exit(0);
		}
	}
	public static void main(String... args)
	{
```

```
            validate();
            System.out.printf("Summary :: PROCESSOR_COUNT %s, Running Testwith %s of
Threads. %n", PROCESSOR_COUNT, PROCESSOR_COUNT);
            long currentMili = System.currentTimeMillis();
            int testCount = 0;
            do
            {
                  reset();
                  for (int i = 0; i < PROCESSOR_COUNT; i++)
                        THREADS[i] = new Thread(SingletoneTest::getInstance);

                  for (int i = 0; i < PROCESSOR_COUNT; i++)
                        THREADS[i].start();

                  for (int i = 0; i < PROCESSOR_COUNT; i++)
                        try
                        {
                              THREADS[i].join();
                        }
                        catch (InterruptedException e)
                        {
                              e.printStackTrace();
                              Thread.currentThread().interrupt();
                        }
                  testCount++;
            }
            while (instancesCount <= 1 && testCount < Integer.MAX_VALUE);

                                    System.out.printf("Singleton Pattern is broken after %d try.
                        %nNumber of instances count is %d. %nTest duration %dms", testCount,
instancesCount, System.currentTimeMillis() - currentMili);
      }
}
```

Output-
Summary :: PROCESSOR_COUNT 4, Running Test with 4 of Threads.Singleton
Pattern is broken after 144 try.
Number of instances count is 2.Test
duration 232ms

Q.2) Write a python program to find all null values in a given dataset & remove them.

```
# importing pandas as pdimport
pandas as pd
# importing numpy as npimport
numpy as np
# dictionary of lists
```

```
dict = {'First Score':[100, 90, np.nan, 95], 'Second Score':
                [30, 45, 56, np.nan],
                'Third Score':[np.nan, 40, 80, 98]}# creating a
dataframe from list
df = pd.DataFrame(dict)
# using isnull() function print("\n isnull()
function ");print(df.isnull())
# filling missing value using fillna() print("\n After filling
null values with 0");print(df.fillna(0))
```

Output-

isnull() function

	First Score	Second Score	Third Score
0	False	False	True
1	False	False	False
2	True	False	False
3	False	True	False

After filling null values with 0

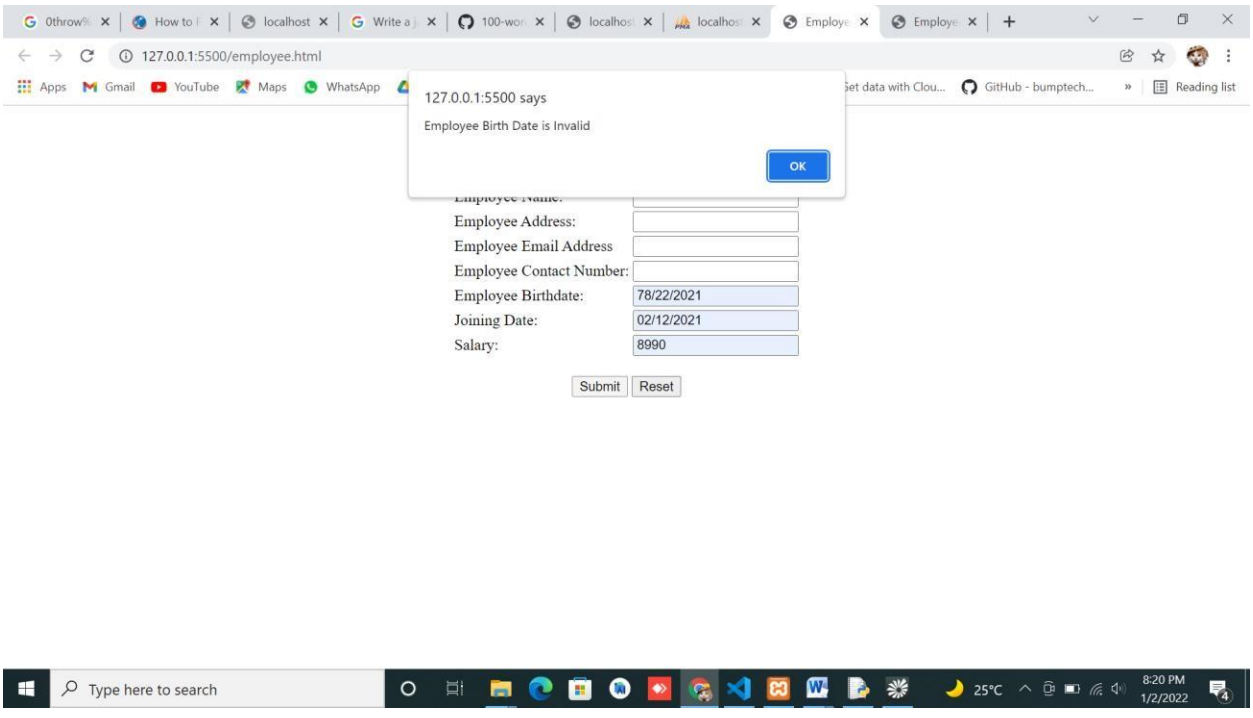
	First Score	Second Score	Third Score
0	100.0	30.0	0.0
1	90.0	45.0	40.0
2	0.0	56.0	80.0
3	95.0	0.0	98.0

Q.3) Create an HTML form that contain the Employee Registration details and write a JavaScript to validate DOB, Joining Date, and Salary.

```
<!DOCTYPE html>
<html lang="en">

<head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Employee Details</title>

</head>

<body>
        <center>
                <h1>Employee Registration Form</h1>
                <form name="frm">
                        <table>
```

```
                        <tr>
                            <td>Employee Name: </td>
                            <td><input type="text" name="name"></td>
                        </tr>
                        <tr>
                            <td>Employee Address: </td>
                            <td><input type="text" name="Address"></td>
                        </tr>

                        <tr>
                            <td>Employee Email Address</td>
                            <td><input type="text" name="Email"></td>
                        </tr>
                        <tr>
                            <td>Employee Contact Number: </td>
                            <td><input type="text" name="Telephone"></td>
                        </tr>
                        <tr>
                            <td>Employee Birthdate: </td>
                            <td><input type="text" name="bdate"
placeholder="dd/mm/yyyy"></td>
                        </tr>
                        <tr>
                            <td>Joining Date: </td>
                            <td><input type="text" name="jdate"
placeholder="dd/mm/yyyy"></td>
                        </tr>
                        <tr>
                            <td>Salary: </td>
                            <td><input type="text" name="salary" ></td>
                        </tr>
                    </table><br>
                    <input type="button" name="b1" onclick="validate()"
value="Submit">
                    <input type="reset" name="b2">
                </form>
        </center>
        <script>
            function validate() {
        var bdate = document.frm.bdate.value; var jdate =
        document.frm.jdate.value; var salary =
        document.frm.salary.value;
        let date = /^(0?[1-9]|[12][0-9]|3[01])[\/\-](0?[1-9]|1[012])[\/\-
]\d{4}$/;
        if (bdate != "" && jdate != "" && salary != "") {
```

```
        if (isNaN(salary)) {
            alert("Enter only digit please");
            } else                  if(bdate.search(date) == -1){
                alert("Employee Birth Date is Invalid");
            }
            else if(jdate.search(date) == -1){ alert("Employee Join
                Date is Invalid");
            }else
              {
                alert("Submit succesfully. ..... ");
            }
        }
        else {
            alert("Invalid fields");
        }
}
        </script>
</body>
</html>
```



Slip-3)

Q.1) Write a java program to implement built-in support (java.util.observable) Weather station with members temperature, humidity, pressure & methods measurementchanged(), setmeasurements(), gettemperature(), gethumidity(), getpressure().

1.Create Interface

Observer.java

```
package javaprograms;
public interface Observer {
	public void update(float temp,float humidity,float pressure);
}
```

Displayelement.java

```
package hello;

public interface DisplayElement {

	public void display();
}
```

Subject.java

```
package hello;

public interface Subject {

	public void registerObserver(Observer o);public void
	removeObserver(Observer o); public void
	notifyObservers();
}
```

2. create classes

CurrentConditionDispaly.java

```
package hello;

public class CurrentConditionDispaly implements
Observer,DisplayElement {

	private float temprature; private float
	humidity; private Subject
	weatherData;

	public CurrentConditionDispaly(Subject weatherData)
{
		this.weatherData=weatherData;
		weatherData.registerObserver(this);
```

```
}

	public void update(float temprature,float humidity,floatpressure) {
		this.temprature=temprature;
		this.humidity=humidity; display();
	}
	public void display()
	{
		System.out.println("current conditions:"+temprature+"Fdegree and
"+humidity+"% humidity");
	}


}
```

ForecastDisplay.java

```
package hello;

public class ForecastDisplay implements Observer,DisplayElement {

	private float   currentpressure=29.92f;
	private float lastpressure;
	private WeatherData weatherData;

	public ForecastDisplay(WeatherData weaherdata) {
		this.weatherData=weatherData;
		weatherData.registerObserver(this);
	}
	public void update(float temp,float humidity,float pressure) {
		lastpressure=currentpressure;
		currentpressure=pressure;
		display();
	}

	public void display()
	{
		System.out.println("forecast:");
		if(currentpressure > lastpressure) { System.out.println("improving weather on the
			way!.");
		}else if(currentpressure==lastpressure) {
			System.out.println("more of the same");
		}else if(currentpressure < lastpressure) {
```

```
                        System.out.println("watch out for cooler,rainy
weather");
                }

        }

}
```

HeatIndexDisplay.java

```
package hello;

public class HeatIndexDisplay implements Observer,DisplayElement {
        float heatIndex=0.0f;
        private WeatherData weatherData;
        public HeatIndexDisplay(WeatherData weatherData) {
                this.weatherData=weatherData;
                weatherData.registerObserver(this);
           }
        public void update(float t,float rh,float pressure) {
                heatIndex=computeHeatIndex(t,rh);
                display();

        }
        private float computeHeatIndex(float t,float rh) {
                float index=(float)((16.923 + (0.185212 * t) + (5.37941 * rh) - (0.100254 * t *
rh) + (0.000345372 *(t * t * rh ))) +(0.00728898
* (rh * rh)) + (0.000345372 * (t * t * rh))   - (0.000814971 * (t * rh
* rh))+ (0.0000102102 * (t * t * rh * rh)) -(0.000038646 * (t * t * t)) + (0.0000291683 * (rh *
rh * rh)) + (0.00000142721 * (t * t * t *rh )) + (0.000000197483 * (t * rh * rh * rh)) -
(0.000000218429 * (t *
t * t * rh * rh )) + (0.000000000843296 * (t * t * rh * rh * rh)) -(0.0000000000481975 * (t *
t * t * rh * rh * rh)));
        return index;
        }
        public void display()
        {
                System.out.println("heat index"+heatIndex);
        }
}
```

StatisticDisplay.java

```
package hello;

public class StatisticDisplay   implements Observer,DisplayElement {
          private float maxTemp=0.0f;
          private float minTemp=200;
          private float tempSum=0.0f;
```

```
    private int numReadings;
  private WeatherData weatherData;
  public StatisticDisplay(WeatherData weatherData) {
    this.weatherData=weatherData;
    weatherData.registerObserver(this);
  }

  public void update(float temp,float humidity,float pressure)
  {
    tempSum=temp;
    numReadings++;

    if(temp > maxTemp)
    {
        maxTemp=temp;
   }
    if(temp < minTemp) {
        minTemp=temp;
    }
    display();
  }

  public void display()
  {
    System.out.println("AVG?MIN?MAX
temprature="+(tempSum/numReadings )+"/"+maxTemp+"/"+minTemp);

  }
}
```

WeatherData.java

```
package hello;

import java.util.ArrayList;

public class WeatherData implements Subject{private
  ArrayList<Observer> observers; private float
  temprature;
  private   float humidity;
  private float pressure;


  public WeatherData() { observers=new
      ArrayList<>();
  }
```

```
	public void registerObserver(Observer o) {
		observers.add(o);
	}

	public void removeObserver(Observer o) {int
		i=observers.indexOf(o); if(i>=0) {
			observers.remove(i);
		}
	}

public void notifyObservers() {
	for(int i=0;i<observers.size();i++) {
		Observer observer=(Observer)observers.get(i);
		observer.update(temprature, humidity, pressure);
	}

}
public void measurementChanged() {
	notifyObservers();
}
public void setMeasurement(float temprature,float humidity,floatpressure) {
this.temprature=temprature;
this.humidity=humidity;
this.pressure=pressure;
measurementChanged();
}
public float getTemprature()
{
   return temprature;
}
public float gethumidity()
{
return   humidity;
}
public float getpressure()
{
return pressure;
}
}
```

WeatherStation.java

```
package hello; import
java.io.*;
public class WeatherStation {
```

```
                public static void main(String[] args) {
                 // TODO Auto-generated method stub
            //try {
            WeatherData weatherData=new WeatherData();
            CurrentConditionDispaly currentDisplay=new
CurrentConditionDispaly(weatherData); StatisticDisplay
            statisticDisplay=new
StatisticDisplay(weatherData);
            weatherData.setMeasurement(80,65,30.4f);
            weatherData.setMeasurement(82, 70,29.2f);
            weatherData.setMeasurement(78,90,29.2f);
        }
}
```

Output-
current conditions:80.0F degree and 65.0% humidity
AVG?MIN?MAX temprature=80.0/80.0/80.0

current conditions:82.0F degree and 70.0% humidity
AVG?MIN?MAX temprature=41.0/82.0/80.0
current conditions:78.0F degree and 90.0% humidity
AVG?MIN?MAX temprature=26.0/82.0/78.0

Q.2) Write a python program to make Categorical values in numeric format for a given dataset

```
#import pandas import
pandas as pd

# read csv file
df = pd.read_csv('Customers.csv')print(df)
print("\n After repalcing Category Male as 0 and Female as 1");# replacing values

df['GENDER'].replace(['Male', 'Female'],
                                        [0, 1], inplace=True)
print(df)
```

Output:

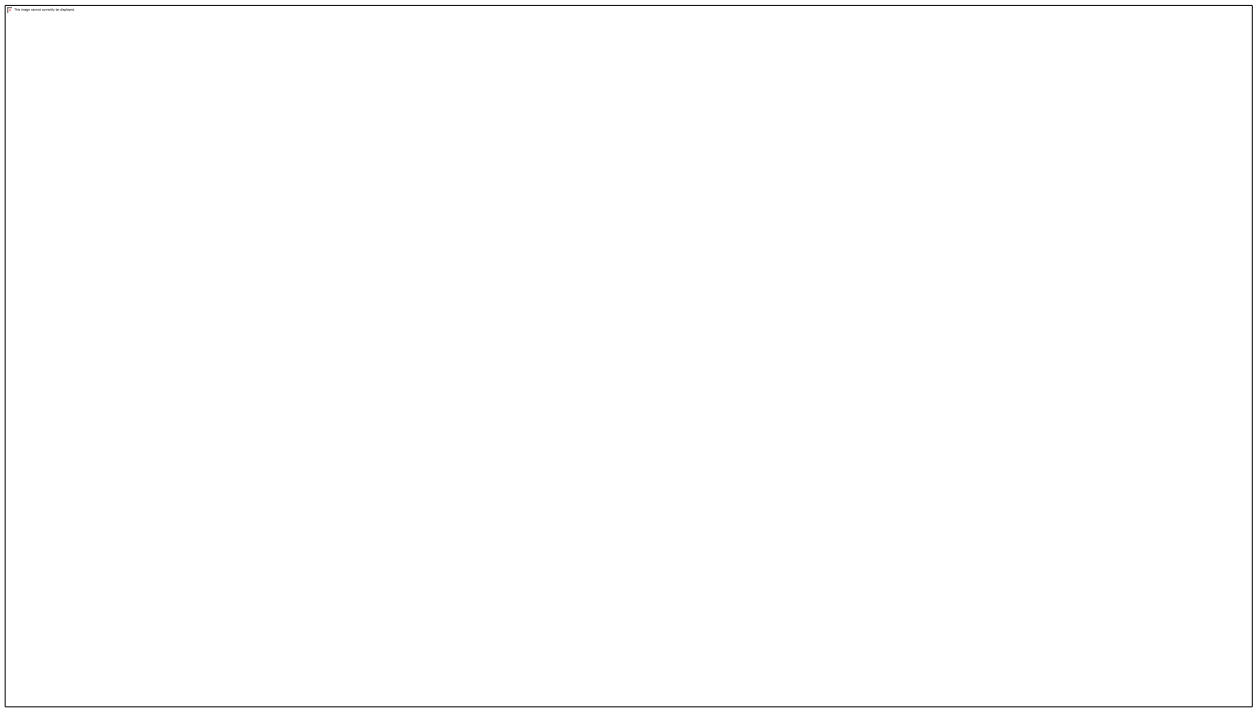
```
Name  Episodes  Gender
0 Sheldon   42  male
1  Penny   24  female
2   Amy    31 female
3  Penny   29 female
4   Raj   37  male
5 Sheldon   40  male
   Name  Episodes  Gender  female  male
```

```
0 Sheldon   42  male   0  1
1  Penny    24  female 1  0
2   Amy     31  female 1  0
3  Penny    29  female 1  0
4   Raj    37  male   0  1
5 Sheldon   40  male   0  1
```

Q.3) Create an HTML form for Login and write a JavaScript to validate email ID using Regular Expression.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Login Page</title>
</head>
<body>
   <center>
      <h1>Login Form</h1>
      <form name="frm">
         <table>
            <tr>
               <td>User Name</td>
               <td><input type="text" name="Email"></td>
            </tr>
            <tr>
               <td>Password </td>
               <td><input type="text" name="password"></td>
            </tr>

         </table><br>
         <input type="button" name="b1" onclick="validate()" value="Submit">
         <input type="reset" name="b2">
      </form>
   </center>
   <script>
      function validate() {
         var email = document.frm.Email.value;
         var password = document.frm.password.value;
         var filter = /^([a-zA-Z0-9_\.\-])+\@(([a-zA-Z0-9\-])+\.)+([a-zA-Z0-9]{2,4})+$/;
         if (email != "" && password != "") {
            if (!email.match(filter)) {
               alert("Invalid email fields");
            } else if (password.length < 6 || password.length > 8) {
               alert("Password min and max length is 6.");
```

```
            }
            else {
                alert("Thank you for Login")
            }
        }
        else {
            alert("Invalid fields");
        }
    }
  </script>
</body>
</html>
```



Slip-4)

Q.1) Write a Java Program to implement Factory method for Pizza Store with createPizza(), orederPizza(), prepare(), Bake(), cut(), box(). Use this to create variety of pizzas like NyStyleCheesePizza, ChicagoStyleCheesePizza etc.

Create Class –

1)Pizza.class

```
package javaprograms;
import java.util.ArrayList;

abstract public class Pizza {String
        name;
        String dough;
        String sauce;
```

```
ArrayList toppings = new ArrayList();

public String getName() {
        return name;
}

public void prepare() { System.out.println("Preparing " +
        name);
}

public void bake() { System.out.println("Baking " +
        name);
}

public void cut() { System.out.println("Cutting " +
        name);
}

public void box() { System.out.println("Boxing " +
        name);
}

public String toString() {
        // code to display pizza name and ingredientsStringBuffer
        display = new StringBuffer(); display.append("---- " + name
        + " ---------------------------------------------------- \n");
        display.append(dough + "\n");
        display.append(sauce + "\n");
        for (int i = 0; i < toppings.size(); i++) { display.append((String) toppings.get(i) +
                "\n");
        }
        return display.toString();
```

```
	}
}
```
2)PizzaStore.class
```
package javaprograms;

public class PizzaStore {
	SimplePizzaFactory factory;

	public PizzaStore(SimplePizzaFactory factory) {
		this.factory = factory;
	}

	public Pizza orderPizza(String type) {Pizza pizza;

		pizza = factory.createPizza(type);

		pizza.prepare();
		pizza.bake();
		pizza.cut();
		pizza.box();

		return pizza;
	}
}
```
3)SimplePizzaFactory.class
```
package javaprograms;

public class SimplePizzaFactory {

	public Pizza createPizza(String type) {Pizza pizza =
		null;

		if (type.equals("cheese")) { pizza = new
			NYCheesePizza();
		} else if (type.equals("veggie")) { pizza = new
			ChicagoCheesePizza();
		}
		return pizza;
	}
}
```
4)NYCheesePizza.class
```
package javaprograms;
public class NYCheesePizza extends Pizza {
	public   NYCheesePizza()   { name =
		"NY Cheese Pizza";
```

```
            dough = "Regular Crust";
            sauce = "Marinara Pizza Sauce";
            toppings.add("Fresh Mozzarella");
            toppings.add("Parmesan");
        }
}
```

5)ChicagoCheesePizza.class

```
package javaprograms;
public class ChicagoCheesePizza extends Pizza {
        public   ChicagoCheesePizza()   { name =
            "Chicago Cheese Pizza";dough =
            "Crust";
            sauce = "Marinara sauce";
            toppings.add("Shredded mozzarella");
            toppings.add("Grated parmesan");
            toppings.add("Diced onion");
            toppings.add("Sliced mushrooms");
            toppings.add("Sliced red pepper");
            toppings.add("Sliced black olives");
        }
}
```

Output-

Preparing NY Cheese Pizza
Baking NY Cheese Pizza Cutting
NY Cheese Pizza Boxing NY
Cheese Pizza
We ordered a NY Cheese Pizza

Preparing Chicago Cheese PizzaBaking
Chicago Cheese Pizza Cutting Chicago
Cheese Pizza Boxing Chicago Cheese
Pizza
We ordered a Chicago Cheese Pizza

Q.2) Write python program to implement simple linear regression for predicting house price.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pltimport
seaborn as sns


#sns.set_style("whitegrid")
#plt.style.use("fivethirtyeight")
```

```
USAhousing = pd.read_csv('USA_Housing.csv')
USAhousing.head()

X = USAhousing[['Avg. Area Income', 'Avg. Area House Age', 'Avg. AreaNumber of Rooms',
                'Avg. Area Number of Bedrooms', 'Area Population']]y =
USAhousing['Price']

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.4,
random_state=101)
from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression(normalize=True)
lin_reg.fit(X_train,y_train)


pred = lin_reg.predict(X_test)
plt.scatter(y_test, pred) plt.show()
```

Q.3) Create nodejs file that will convert the output 'Hello World!' into uppercase letters.

```
var http= require('http'); var
uc=require('upper-case');
http.createServer(function(req,res){ res.writeHead(200,{'Content-
        type':'text/html'});res.write(uc.upperCase("Hello World"));
        res.end();
}).listen(8080);
```

Output-
HELLO WORLD

Slip-5)

Q.1) Write a Java Program to implement Adapter pattern for Enumeration iterator.

```
import java.util.Enumeration;
import java.util.Vector;

public class EnumProduct
```

```
{
	private Vector product;

	public EnumProduct(){
		product = new Vector();
		setProduct("ProductA");
		setProduct("ProductB");
		setProduct("ProductC");
	}

	public void setProduct(String s){
		 product.add(s);
	}

	public Enumeration getProduct(){
		 Enumeration eProduct = product.elements();
		 return eProduct;
	}
}


import java.util.Iterator;
import java.util.Enumeration;
public class EnumToIteratorAdapter implements Iterator
{
	Enumeration enumA;

	public EnumToIteratorAdapter(Enumeration e){
		enumA = e;
	}

	public boolean hasNext(){
		return enumA.hasMoreElements();
	}

	public Object next(){
		return enumA.nextElement();
	}
```

```
        public void remove(){
                throw new UnsupportedOperationException();
        }
}


import java.util.Enumeration;
import java.util.Vector;
import java.util.Iterator;

public class Product
{
        public void displayProduct(Iterator iterator){
                for (; iterator.hasNext();)
                        System.out.println(iterator.next());
        }

         public static void main(String[] args) {
                Product product = new Product();
                EnumProduct enumProduct = new EnumProduct();
                EnumToIteratorAdapter enumToIteratorAdapter = new
EnumToIteratorAdapter(enumProduct.getProduct());
                product.displayProduct(enumToIteratorAdapter);
         }


}
```

Output:

```
ProductA
ProductB
ProductC
```

Q.2) Write a python program to implement multiple Linear Regression for a given dataset.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import pylab as pl

df = pd.read_csv('Fuelconsumption.csv')
df.head()
cdf =
df[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_CITY','FUELCONSUMPTION_H
WY','FUELCONSUMPTION_COMB','CO2EMISSIONS']]
cdf.head()
```

```
plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS, color='blue')
plt.xlabel('Engine Size')
plt.ylabel('Emissions')
plt.show()

msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]

from sklearn import linear_model
regr = linear_model.LinearRegression()x
=
np.asanyarray(train[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']]
)
y = np.asanyarray(train[['CO2EMISSIONS']])
regr.fit(x,y)

print('Coefficients: ', regr.coef_)
```

Q.3) Using nodejs create a web page to read two file names from user and append contents of first file into second file.

```
const fs = require('fs');

// open destination file for appending
var write = fs.createWriteStream("message.txt", {flags: 'a'});
// open source file for reading
var read = fs.createReadStream("input.txt");
write.on('close', function() {
    console.log("done writing");
});
read.pipe(write);
```

Slip-6)

Q.1) Write a Java Program to implement command pattern to test Remote Control.

1) Create Interface-
Command.java

```
package javaprograms;

interface Command
{
    public void execute();
}
```

2) Create Class
Light.java

```
package javaprograms;
public class Light
{
	public void on()
	{
		System.out.println("Light is on");
	}
	public void off()
	{
		System.out.println("Light is off");
	}
}
```

LightOnCommand.java

```
package javaprograms;
class LightOnCommand implements Command
{
	Light light;

	// The constructor is passed the light it
	// is going to control.
	public LightOnCommand(Light light)
	{
	    this.light = light;
	}
	public void execute()
	{
	    light.on();
	}
}
```

LightOffCommand.java

```
package javaprograms;
class LightOffCommand implements Command
{
        Light light;
        public LightOffCommand(Light light)
        {
                this.light = light;
        }
        public void execute()
        {
                 light.off();
        }
}
```

Stereo.java

```
package javaprograms;

public class Stereo
{
        public void on()
        {
                        System.out.println("Stereo  is on");
        }
        public void off()
        {
                        System.out.println("Stereo  is off");
        }
        public void setCD()
        {
                        System.out.println("Stereo  is set " +
                                        "for CD  input");
        }
        public void setDVD()
        {
                        System.out.println("Stereo  is set"+
                                        " for DVD  input");
        }
        public void setRadio()
        {
                        System.out.println("Stereo  is set" +
                                        " for Radio");
        }
        public void setVolume(int volume)
        {
```

```
            // code to set the volume System.out.println("Stereo
            volume set"
                                        + " to " + volume);
        }
}
```

StereoOffCommand.java

```
package javaprograms;

class StereoOffCommand implements Command
{
        Stereo stereo;
        public StereoOffCommand(Stereo stereo)
        {
              this.stereo = stereo;
        }
        public void execute()
        {
            stereo.off();
        }
}
```

StereoOnWithCDCommand.java

```
package javaprograms;

class StereoOnWithCDCommand implements Command
{
         Stereo stereo;
         public StereoOnWithCDCommand(Stereo stereo)
         {
               this.stereo = stereo;
         }
         public void execute()
         {
               stereo.on(); stereo.setCD();
               stereo.setVolume(11);
         }
}
```

SimpleRemoteControl.java

```
package javaprograms; class
SimpleRemoteControl
{
        Command slot;   // only one button
```

```
        public SimpleRemoteControl()
        {
        }

        public void setCommand(Command command)
        {
                // set the command the remote will
                // execute
                slot = command;
        }
        public void buttonWasPressed()
        {
                slot.execute();
        }
}
```

RemoteControlTest.java

```
package javaprograms;

class RemoteControlTest
{
        public static void main(String[] args)
        {
                SimpleRemoteControl remote =
                                new SimpleRemoteControl(); Light
                light = new Light();
                Stereo stereo = new Stereo();

                // we can change command dynamically
                remote.setCommand(new
                                        LightOnCommand(light));
                remote.buttonWasPressed();
                remote.setCommand(new
                                StereoOnWithCDCommand(stereo));
                remote.buttonWasPressed(); remote.setCommand(new
                                        StereoOffCommand(stereo));
                remote.buttonWasPressed();
          }
    }
```

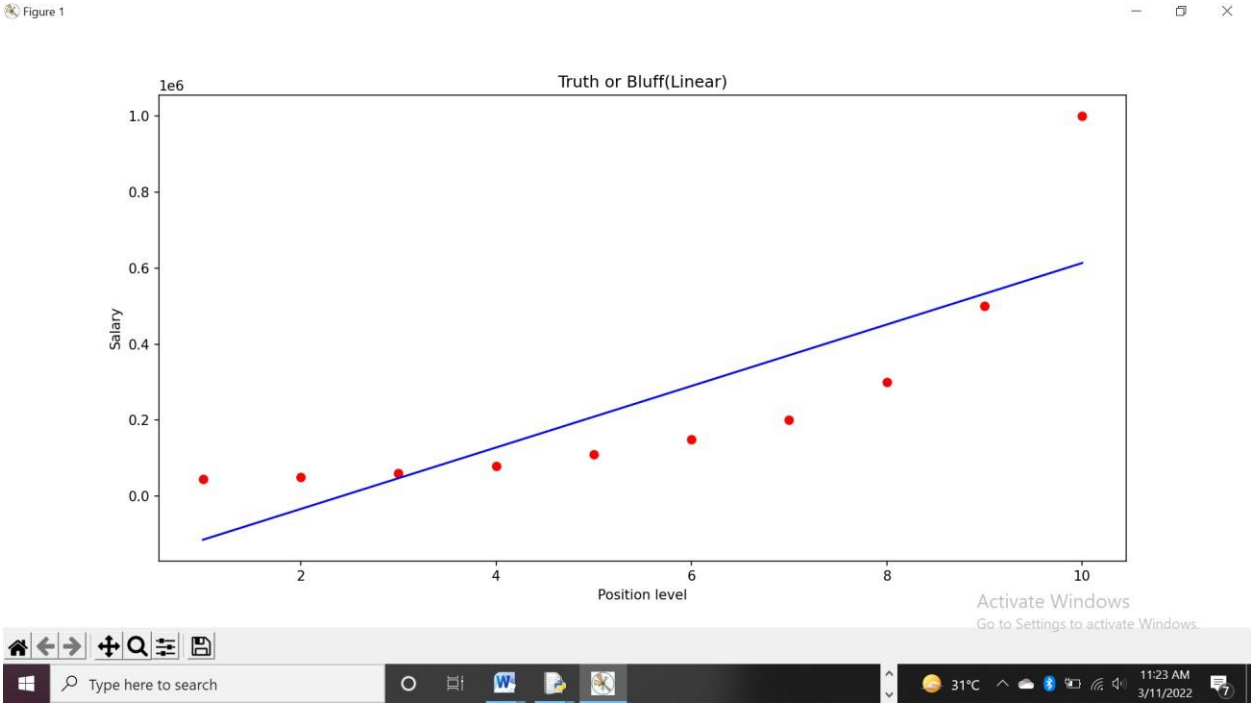
Output- Light is
on Stereo is on
Stereo is set for CD input Stereo volume set
to 11

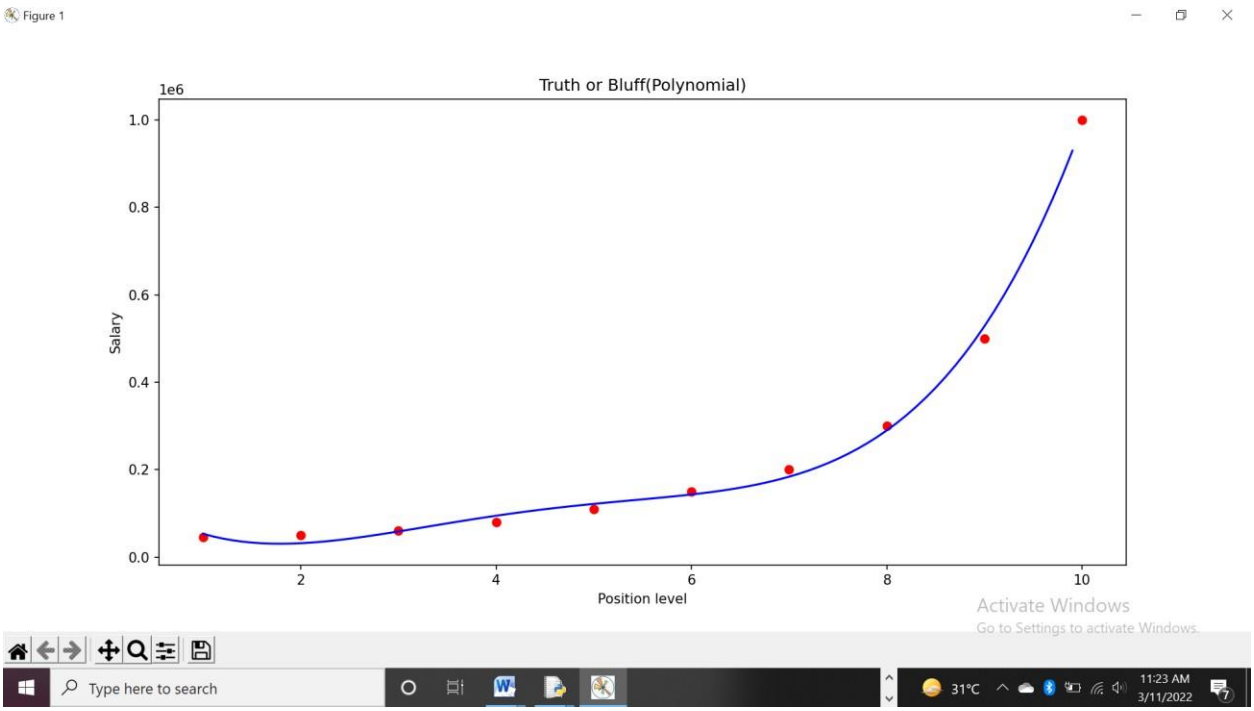
Stereo is off

Q.2) Write a python program to implement Polynomial Regression for given dataset.

```
import numpy as np
import matplotlib.pyplot as plt import pandas as
pd

dataset = pd.read_csv('position_salaries.csv') dataset

X = dataset.iloc[:,1:2].values y =
dataset.iloc[:,2].values

# fitting the linear regression model
from sklearn.linear_model import LinearRegression lin_reg =
LinearRegression()
lin_reg.fit(X,y)

# visualising the linear regression model plt.scatter(X,y, color='red')
plt.plot(X, lin_reg.predict(X),color='blue') plt.title("Truth or
Bluff(Linear)") plt.xlabel('Position level') plt.ylabel('Salary')
plt.show()

# polynomial regression model
from sklearn.preprocessing import PolynomialFeatures poly_reg =
PolynomialFeatures(degree=2)
X_poly = poly_reg.fit_transform(X) X_poly

                # prints X_poly

lin_reg2 = LinearRegression()
lin_reg2.fit(X_poly,y)


# visualising polynomial regression
from sklearn.preprocessing import PolynomialFeatures poly_reg =
PolynomialFeatures(degree=4)
X_poly = poly_reg.fit_transform(X) lin_reg2 =
LinearRegression() lin_reg2.fit(X_poly,y)

X_grid = np.arange(min(X),max(X),0.1)
```

```
X_grid = X_grid.reshape(len(X_grid),1) plt.scatter(X,y, color='red')

plt.plot(X_grid, lin_reg2.predict(poly_reg.fit_transform(X_grid)),color='blue') plt.title("Truth or

Bluff(Polynomial)")
plt.xlabel('Position level')
plt.ylabel('Salary') plt.show()
```





Q.3) Create a Nodejs file that opens the requested file and returns the content to the client . If anything goes wrong throw 404 error.

Fileprogram.js

```
var http = require('http'); var url =
require('url'); var fs = require('fs');

 http.createServer(function (req, res) { var  q  =
    url.parse(req.url,  true); var filename = "." +
    q.pathname;
    fs.readFile(filename, function(err, data) { if (err) {
            res.writeHead(404, {'Content-Type': 'text/html'}); return res.end("404 Not
            Found");
        }
        res.writeHead(200, {'Content-Type': 'text/html'}); res.write(data);
        return res.end();
    });
}).listen(8080);
```

summer.html

```
<!DOCTYPE html>
<body>
<h1>Summer</h1>
```

```
<p>I love the sun!</p>
</body>
</html>
```

winter.html

```
<!DOCTYPE html>
<body>
<h1>Winter</h1>
<p>I love the snow!</p>
</body>
</html>
```

Slip-7)

Q.1) Write a Java Program to implement undo command to test Ceiling fan.

SimpleRemoteControl class

```
class SimpleRemoteControl
{
 Command slot;

 public SimpleRemoteControl()
 {
 }

 public void setCommand(Command command)
 {
 slot = command;
 }
 public void buttonWasPressed()
 {
 slot.execute();
 }
}
```

CeilingFan class

```
public class CeilingFan {
public static final int HIGH = 3;
public static final int MEDIUM = 2;
public static final int LOW = 1;
public static final int OFF = 0;
String location;
int speed;
public CeilingFan(String location) {
this.location = location;
speed = OFF; }
public void high() {
speed = HIGH;
```

```
System.out.println(location + " ceiling fan is on high"); }
public void medium() {
speed = MEDIUM;
System.out.println(location + " ceiling fan is on medium"); }
public void low() {
speed = LOW;
System.out.println(location + " ceiling fan is on low"); }
public void off() {
speed = OFF;
System.out.println(location + " ceiling fan is off"); }
public int getSpeed() {
```

```
return speed; }
}
```
CeilingFanHighCommand class
```
public class CeilingFanHighCommand implements Command {
CeilingFan ceilingFan;
int prevSpeed;
public CeilingFanHighCommand(CeilingFan ceilingFan) {
this.ceilingFan = ceilingFan;
}
public void execute() {
prevSpeed = ceilingFan.getSpeed();
ceilingFan.high();
}
public void undo() {
if (prevSpeed == CeilingFan.HIGH) {
ceilingFan.high();
} else if (prevSpeed == CeilingFan.MEDIUM) {
ceilingFan.medium();
} else if (prevSpeed == CeilingFan.LOW) {
ceilingFan.low();
} else if (prevSpeed == CeilingFan.OFF) {
ceilingFan.off();
}
}
}
```
Command interface
```
public interface Command {
        public void execute();
        public void undo ();
}
```

RemoteControl class
```
public class RemoteControl {
Command[] onCommands;
Command[] offCommands;
Command undoCommand;
public RemoteControl() {
onCommands = new Command[7];
offCommands = new Command[7];
Command noCommand = new NoCommand();
for (int i = 0; i < 7; i++) {
onCommands[i] = noCommand;
offCommands[i] = noCommand;
}
undoCommand = noCommand;
}
```

```
public void setCommand(int slot, Command onCommand, Command offCommand) {
onCommands[slot] = onCommand;
offCommands[slot] = offCommand;
}
public void onButtonWasPushed(int slot) {
onCommands[slot].execute();
undoCommand = onCommands[slot];
}
public void offButtonWasPushed(int slot) {
offCommands[slot].execute();
undoCommand = onCommands[slot];
}
public void undoButtonWasPushed ( ) {
undoCommand.undo ( );
}
}
```

RemoteContoleTest class

```
class RemoteControlTest
{
 public static void main(String[] args)
 {
 RemoteControl remote =new RemoteControl();
 CeilingFan CeilingFan = new CeilingFan("center");
 remote.setCommand(0, new CeilingFanHighCommand(CeilingFan), new
CeilingFanHighCommand(CeilingFan));
 remote.onButtonWasPushed(0);
 remote.offButtonWasPushed(0);
 remote.undoButtonWasPushed();
 }
 }
```

Output :

```
center ceiling fan is on high
center ceiling fan is on high
center ceiling fan is off
```

Q.2) Write a python program to Implement Naïve Bayes.

```
from sklearn import datasets
from sklearn import metrics
from sklearn.naive_bayes import GaussianNB

dataset = datasets.load_iris()

#Creating our Naive Bayes Model
model = GaussianNB()
model.fit(dataset.data, dataset.target)
```

```
#Making Predictions
expected = dataset.target
predicted = model.predict(dataset.data)

#Getting Accuracy and Statistics
print(metrics.classification_report(expected, predicted))
print(metrics.confusion_matrix(expected, predicted))
```

3) Create Nodejs file that write an HTML form with an upload fields.

```
var http = require('http');
var formidable = require('formidable');

http.createServer(function (req, res) {
  if (req.url == '/fileupload') {
    var form = new formidable.IncomingForm();
    form.parse(req, function (err, fields, files) {
      res.write('File uploaded');
      res.end();
    });
  } else {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write('<form action="fileupload" method="post"
enctype="multipart/form-data">');
    res.write('<input type="file" name="filetoupload"><br>');
    res.write('<input type="submit">');
    res.write('</form>');
    return res.end();
  }
}).listen(8080);
```

Slip-9)

Q.1) Design simple HR Application using Spring Framework.

Q.2) Write a python program to implement linear SVM.

```
# Import the Libraries import
numpy as np
import matplotlib.pyplot as plt from sklearn
import svm, datasets

# Import some Data from the iris Data Set iris =
datasets.load_iris()

# Take only the first two features of Data.
# To avoid the slicing, Two-Dim Dataset can be used
```

```
X = iris.data[:, :2] y =
iris.target

# C is the SVM regularization parameter C = 1.0

# Create an Instance of SVM and Fit out the data.
# Data is not scaled so as to be able to plot the support vectors svc = svm.SVC(kernel
='linear', C = 1).fit(X, y) #Fit the SVM model according to the given training data.
                                                                    #SVC=Support Vector
Classifier

# create a mesh to plot
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1 h = (x_max /
x_min)/100
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),                    #Return
coordinate matrices from coordinate vectors.
              np.arange(y_min, y_max, h))

# Plot the data for Proper Visual Representation plt.subplot(1, 1, 1)
#Add a subplot to the current figure.subplot(nrows, ncols,
indexOfSubplot)

# Predict the result by giving Data to the model
Z = svc.predict(np.c_[xx.ravel(), yy.ravel()])                    #ravel()-Return a
contiguous flattened array.
Z = Z.reshape(xx.shape)
```

```
plt.contourf(xx, yy, Z, cmap = plt.cm.Paired, alpha = 0.8) #contour and contourf draw contour lines and filled contours.

plt.scatter(X[:, 0], X[:, 1], c = y, cmap = plt.cm.Paired) plt.xlabel('Sepal length')
plt.ylabel('Sepal width') plt.xlim(xx.min(), xx.max())
plt.title('SVC with linear kernel')

# Output the Plot plt.show()
```

Q.3) Create a node.js file that Select all records from the "customers" table, and display the result object on console.

```
var mysql = require('mysql');

var con = mysql.createConnection({ host:
   "localhost",
   user: "root",
   password: "", database: "mydb"
});

con.connect(function(err) { if (err) throw
   err;
   con.query("SELECT * FROM customers", function (err, result, fields)
{
      if (err) throw err; console.log(result);
   });
});
```

Slip-10)

Q.1) Write a java program to implement strategy pattern for duck behavior create instance variable that holds current state of duck from there we just need to handle all flying behavior and quack behavior.

1) Create Interface

QuackBehaviour.java

```
package javaprograms;
public interface QuackBehaviour {
            public default void quack() {
               System.out.println("Quack");
            }
}
```

FlyBehaviour.java

```
package javaprograms;

public interface FlyBehaviour {
            public void fly();
}
```

2)Create Class -

FlyWithWings.java

```
package javaprograms;
public class FlyWithWings implements FlyBehaviour {
        public void fly() { System.out.println("I'm
            flying!!");
        }
}
```

Quack.java

```
package javaprograms;

public class Quack implements QuackBehaviour {
        public void quack() {
            System.out.println("Quack");
        }
}
```

ModolDuck.java

```
package javaprograms;
public class ModolDuck extends Duck {
        public ModolDuck() { flyBehaviour = new
            FlyNoWay();quackBehaviour = new
            Quack();
```

```
        }

        public void display() { System.out.println("I'm a model
            duck");
        }
}
```

MallardDuck.java

```
package javaprograms;

public class MallardDuck extends Duck {

        public MallardDuck() { quackBehaviour = new
            Quack(); flyBehaviour = new
            FlyWithWings();
        }

        public void display() {
            System.out.println("I'm a real Mallard duck");
        }
}
```

Duck.java

```
package javaprograms;

public class MallardDuck extends Duck {

        public MallardDuck() { quackBehaviour = new
            Quack(); flyBehaviour = new
            FlyWithWings();
        }

        public void display() {
            System.out.println("I'm a real Mallard duck");
        }
}
```

FlyRocketPowered.java

```
package javaprograms;
public class FlyRocketPowered implements FlyBehaviour {
        public void fly() {
            System.out.println("I'm flying with a rocket!");
        }
}
```

FlyNoWay.java

```
package javaprograms;
public class FlyNoWay implements FlyBehaviour {
        public void fly() { System.out.println("I can't
            fly");
        }
}
```

MiniDuckSimulator.java

```
package javaprograms;

public class MiniDuckSimulator {
        public static void main(String[] args) {Duck mallard
            = new MallardDuck(); mallard.performQuack();
            mallard.performFly();
            Duck model = new ModolDuck();

            model.performFly();

            model.setFlyBehaviour(new FlyRocketPowered());

            model.performFly();
        }
}
```

Output-

```
Quack
I'm flying!!
I can't fly
I'm flying with a rocket!
```

Q. 2) Write a Python program to prepare Scatter Plot for Iris Dataset.

```
import pandas as pd
import matplotlib.pyplot as plt
iris=pd.read_csv("Iris.csv")
iris.plot(kind="scatter",x='SepalLengthCm',y='PetalLengthCm')plt.grid()
plt.show()
```

Q.3) Create a node.js file that Insert Multiple Records in "student" table, and display the result object on console.

```
var mysql = require('mysql');

var con = mysql.createConnection({host:
    "localhost",
```

```
    user: "root",
    password: "",
    database: "mydb"
});
con.connect(function(err) { if (err)
    throw err;
    console.log("Connected!");
    var sql = "INSERT INTO customers (name, address) VALUES ?";var values =
    [
        ['John', 'Highway 71'],
        ['Peter', 'Lowstreet 4'],
        ['Amy', 'Apple st 652'],
        ['Hannah', 'Mountain 21'],
        ['Michael', 'Valley 345'],
        ['Sandy', 'Ocean blvd 2'],
        ['Betty', 'Green Grass 1'],
        ['Richard', 'Sky st 331'],
        ['Susan', 'One way 98'],
        ['Vicky', 'Yellow Garden 2'],
        ['Ben', 'Park Lane 38'], ['William',
        'Central st 954'],['Chuck', 'Main Road
        989'],
        ['Viola', 'Sideway 1633']
    ];
    con.query(sql, [values], function (err, result) {if (err) throw err;
        console.log("Number of records inserted: " + result.affectedRows);
    });
});
```

Slip-11)

Q.1) Write a java program to implement Adapter pattern to design Heart Model to Beat Model.

Q. 2) Write a python program to find all null values in a given dataset and remove them.

```
# importing pandas as pdimport
pandas as pd

# importing numpy as npimport
numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95], 'Second Score':
                [30, 45, 56, np.nan],
                'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe from listdf =
pd.DataFrame(dict)
```

```
# using isnull() function print("\n isnull()
function ");print(df.isnull())

# filling missing value using fillna() print("\n After filling
null values with 0");print(df.fillna(0))
```

Q.3) Create a node.js file that Select all records from the "customers" table, and delete the specified record.

```
var mysql = require('mysql');

var con = mysql.createConnection({host:
    "localhost",
    user: "root",
    password: "",
    database: "mydb"
});

con.connect(function(err) {if (err)
    throw err;
    var sql = "DELETE FROM customers WHERE address =
'Shivajinagar,Sangamner'";
```

```
    con.query(sql, function (err, result) {if (err) throw err;
        console.log("Number of records deleted: " + result.affectedRows);
    });
});
```

Slip-12)

Q.1) Write a java program to implement decorator pattern for interface car to define the assemble() method & then decorate it to sports car & luxury car.

DecoratorPatternTest class

```
package decorator;

public class DecoratorPatternTest {

	public static void main(String[] args) {
		Car sportsCar = new SportsCar(new BasicCar());
		sportsCar.assemble();
		System.out.println("\n***");

		Car LuxuryCar = new SportsCar(new LuxuryCar(new BasicCar()));
		LuxuryCar.assemble();
	}

}
```

Car interface

```
package decorator;

public interface Car {

	public void assemble();
}
```

BasicCar class

```
package decorator;

public class BasicCar implements Car {

	@Override
	public void assemble() {
		System.out.print("Basic Car.");
	}

}
```

CarDecorator class

```
package decorator;

public class CarDecorator implements Car {
```

```
protected Car car;

public CarDecorator(Car c){
        this.car=c;
```

```
	}

	@Override
	public void assemble() {
		this.car.assemble();
	}

}
```

LuxuryCar class

```
package decorator;

public class LuxuryCar extends CarDecorator {

	public LuxuryCar(Car c) {
		super(c);
	}

	@Override
	public void assemble(){
		super.assemble();
		System.out.print(" Adding features of Luxury Car.");
	}
}
```

SportsCar class

```
package decorator;

public class SportsCar extends CarDecorator {

	public SportsCar(Car c) {
		super(c);
	}

	@Override
	public void assemble(){
		super.assemble();
		System.out.print(" Adding features of Sports Car.");
	}
}
```

Output:

```
Basic Car. Adding features of Sports Car.
***
Basic Car. Adding features of Luxury Car. Adding features of Sports Car.
```

Q.2) Write a python program to make Categorical values in numeric format for a given dataset

```
#import pandas
```

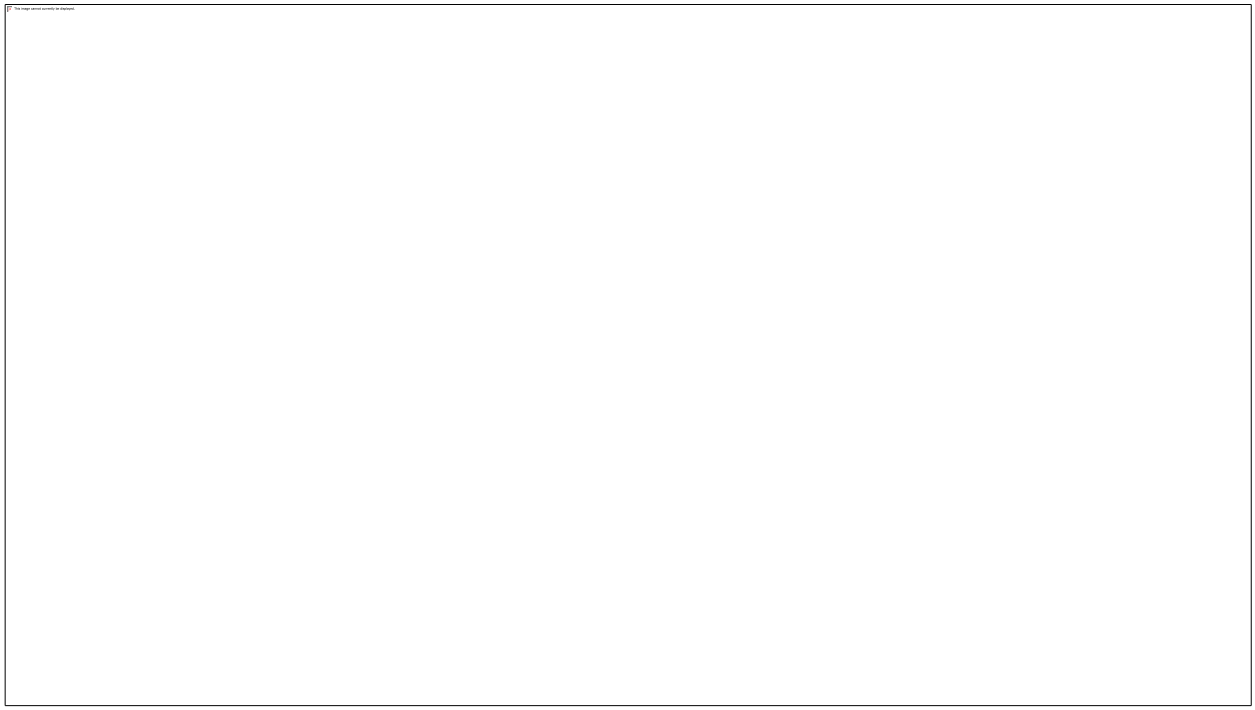
```
import pandas as pd

# read csv file
df = pd.read_csv('Customers.csv')
print(df)
print("\n After repalcing Category Male as 0 and Female as 1");
# replacing values

df['GENDER'].replace(['Male', 'Female'],
                        [0, 1], inplace=True)
print(df)
```

Output:

```
Name  Episodes  Gender
0 Sheldon   42  male
1  Penny    24  female
2   Amy     31  female
3  Penny    29  female
4   Raj    37  male
5 Sheldon   40  male
   Name  Episodes  Gender  female  male
0 Sheldon   42  male   0  1
1  Penny    24  female  1  0
2   Amy     31  female  1  0
3  Penny    29  female  1  0
4   Raj    37  male   0  1
5 Sheldon   40  male   0  1
```

Q.3) Create a Simple Web Server using node js.

```
var http = require('http');
//create a server object:
http.createServer(function (req, res) {
  res.write('Hello World!'); //write a response to the client
  res.end(); //end the response
}).listen(8080); //the server object listens on port 8080
```



Q.1) Write a java program to implement an adapter design pattern in mobile charger. Define two classes Volt (to measure volts) and Socket (producing constant volts of 120v). Build an adapter that can produce 3 volts, 12 volts and default 120 volts. Implement adapter pattern using class adapter.

AdapterPatternTest class

```
package Adapter;

public class AdapterPatternTest {

	public static void main(String[] args) {

testClassAdapter();
testObjectAdapter();
	}

	private static void testObjectAdapter() {
SocketAdapter sockAdapter = new SocketObjectAdapterImpl(); Volt v3 =
getVolt(sockAdapter,3);
Volt v12 = getVolt(sockAdapter,12); Volt v120 =
getVolt(sockAdapter,120);
System.out.println("v3 volts using Object Adapter="+v3.getVolts()); System.out.println("v12
volts using Object Adapter="+v12.getVolts()); System.out.println("v120 volts using Object
Adapter="+v120.getVolts());
	}

	private static void testClassAdapter() {
SocketAdapter sockAdapter = new SocketClassAdapterImpl(); Volt v3 =
getVolt(sockAdapter,3);
Volt v12 = getVolt(sockAdapter,12); Volt v120 =
getVolt(sockAdapter,120);
```

```
        System.out.println("v3 volts using Class Adapter="+v3.getVolts()); System.out.println("v12
        volts using Class Adapter="+v12.getVolts()); System.out.println("v120 volts using Class
        Adapter="+v120.getVolts());
                }

private static Volt getVolt(SocketAdapter sockAdapter, int i) { switch (i){
        case 3: return sockAdapter.get3Volt(); case 12: return
        sockAdapter.get12Volt();
        case 120: return sockAdapter.get120Volt(); default: return
        sockAdapter.get120Volt();
                        }
                }
        }
```

Socket class

```
package Adapter;
public class Socket {

	public Volt getVolt(){
return new Volt(120);
	}
}
```

SocketAdapter interface

```
package Adapter;

public interface SocketAdapter {

	public Volt get120Volt();

	public Volt get12Volt();

	public Volt get3Volt();
}
```

SocketClassAdapterImpl class

```
package Adapter;

public class SocketClassAdapterImpl extends Socket implements SocketAdapter{

	@Override
public Volt get120Volt() { return getVolt();
	}

	@Override
public Volt get12Volt() { Volt v= getVolt();
	return convertVolt(v,10);
	}

	@Override
	public Volt get3Volt() {
	Volt v= getVolt();
	return convertVolt(v,40);
	}

private Volt convertVolt(Volt v, int i) { return new
	Volt(v.getVolts()/i);
	}

	}
```

SocketObjectAdapterImpl class

```
package Adapter;

public class SocketObjectAdapterImpl implements SocketAdapter{

        //Using Composition for adapter pattern
        private Socket sock = new Socket();

        @Override
public Volt get120Volt() { return
        sock.getVolt();
        }

        @Override
        public Volt get12Volt() {
        Volt v= sock.getVolt(); return
        convertVolt(v,10);
        }

        @Override
        public Volt get3Volt() {
        Volt v= sock.getVolt(); return
        convertVolt(v,40);
        }

private Volt convertVolt(Volt v, int i) { return new
        Volt(v.getVolts()/i);
        }
        }
```

Volt class

```
package Adapter;

public class Volt {

        private int volts;

        public Volt(int v){
        this.volts=v;
        }

        public int getVolts() {
        return volts;
        }

public void setVolts(int volts) { this.volts = volts;
```

```
        }
}
```

Output:
v3 volts using Class Adapter=3 v12
volts using Class Adapter=12
v120 volts using Class Adapter=120
v3 volts using Object Adapter=3 v12
volts using Object Adapter=12
v120 volts using Object Adapter=120

Q.2) Write a Python program to prepare Scatter Plot for Iris Dataset

```
import pandas as pd
import matplotlib.pyplot as plt
iris=pd.read_csv("Iris.csv")
iris.plot(kind="scatter",x='SepalLengthCm',y='PetalLengthCm')
plt.grid()
plt.show()
```

Q.1) Write a java program to implement command design pattern for command interface with execute() use this to create verify of commands for Lighton command, Lightoff command, Doorup command, Storeon with CDCommand.

Create Interface-
Command.java

```
package Command;

public interface Command {
      public void execute();
}
```

Create Class-
Stereo.java

```
package Command;

public class Stereo
{
 public void on()
 {
 System.out.println("Stereo is on");
 }
 public void off()
 {
 System.out.println("Stereo is off");
 }
 public void setCD()
```

```
 {
 System.out.println("Stereo is set " +"for CD
 input");
 }
 public void setVolume(int volume)
 {
 // code to set the volume System.out.println("Stereo
 volume set"
 + " to " + volume);
 }
}
```

Light.java

```
package Command;

public class Light { String
      location = "";

      public Light(String location) {
```

```
        this.location = location;
    }

    public void on() {
        System.out.println(location + " light is on");
    }

    public void off() {
        System.out.println(location + " light is off");
    }
}
```

GarageDoor.java

```
package Command;

public class GarageDoor {
     public void up()
     {
     System.out.println("Garage Door is up");
     }
     public void down()
     {
     System.out.println("Garage Door is down");
     }
}
```

RemoteControl.java

```
package Command;

public class RemoteControl {
    Command[] onCommands;
    Command[] offCommands;

    public RemoteControl() {
        onCommands = new
        Command[7]; offCommands =
        new Command[7];

        for (int i = 0; i < 7; i++) {
            onCommands[i] = () -> { };
            offCommands[i] = () -> { };
        }
    }
    public void setCommand(int slot, Command onCommand, Command offCommand) {
        onCommands[slot] = onCommand;
        offCommands[slot] = offCommand;
```

```
    }
    public void onButtonWasPushed(int slot) {
        onCommands[slot].execute();
    }

    public void offButtonWasPushed(int slot) {
        offCommands[slot].execute();
    }

    public String toString() {
        StringBuffer stringBuff = new StringBuffer();
        stringBuff.append("\n------ Remote Control ------------------- \n");
        for (int i = 0; i < onCommands.length; i++) {
            stringBuff.append("[slot " + i + "] " +
onCommands[i].getClass().getName()
                    + "        " + offCommands[i].getClass().getName() + "\n");
        }
        return stringBuff.toString();
    }
}
```

RemoteControlTest.java

```
package Command;

public class RemoteControlTest {
    public static void main(String[] args) { RemoteControl remote
        = new RemoteControl();

        Light light = new Light("Living Room");
        GarageDoor garageDoor = new GarageDoor();
        Stereo stereo = new Stereo();

        Command stereoOnWithCD = () -> {
            stereo.on(); stereo.setCD();
            stereo.setVolume(11);
        };
        remote.setCommand(0, light::on, light::off);
        remote.setCommand(1, garageDoor::up, garageDoor::down);
        remote.setCommand(2, stereoOnWithCD, stereo::off);

        remote.onButtonWasPushed(0);
        remote.offButtonWasPushed(0);
        remote.onButtonWasPushed(1);
        remote.offButtonWasPushed(1);
```

```
        remote.onButtonWasPushed(2);
        remote.offButtonWasPushed(2);
    }
}
```

Output-

Living Room light is on
Living Room light is off
Garage Door is up Garage
Door is down Stereo is on
Stereo is set for CD input
Stereo volume set to 11
Stereo is off

Q.2) Write a python program to find all null values in a given dataset and remove them.

```
# importing pandas as pd
import pandas as pd

# importing numpy as np
import numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95], 'Second
        Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe from listdf =
pd.DataFrame(dict)

# using isnull() function print("\n
isnull() function ");print(df.isnull())

# filling missing value using fillna() print("\n After
filling null values with 0");print(df.fillna(0))
```

Q.1) Write java program to implement façade design pattern for Home Theater.

Amplifier class

```
package facade;

public class Amplifier {
String description;
Tuner tuner;
DvdPlayer dvd;
CdPlayer cd;

    public Amplifier(String description) {
        this.description = description;
```

```
}

public void on() {
		System.out.println(description + " on");
}

public void off() {
		System.out.println(description + " off");
}

	public void setStereoSound() {
		System.out.println(description + " stereo mode on");
}

public void setSurroundSound() {
		System.out.println(description + " surround sound on (5 speakers, 1 subwoofer)");
}

public void setVolume(int level) {
		System.out.println(description + " setting volume to " + level);
}

	public void setTuner(Tuner tuner) {
		System.out.println(description + " setting tuner to " + dvd);
		this.tuner = tuner;
}

public void setDvd(DvdPlayer dvd) {
		System.out.println(description + " setting DVD player to " + dvd);
		this.dvd = dvd;
}

public void setCd(CdPlayer cd) {
```

```
                System.out.println(description + " setting CD player to " + cd);
                this.cd = cd;
}

        public String toString() {
                return description;
}
}
```

CdPlayer class

```
package facade;

public class CdPlayer {
String description; int
currentTrack; Amplifier
amplifier;String title;

        public CdPlayer(String description, Amplifier amplifier) {
                this.description = description;
                this.amplifier = amplifier;
}

public void on() {
                System.out.println(description + " on");
}

public void off() {
                System.out.println(description + " off");
}

public void eject() {
                title =  null;
                System.out.println(description + " eject");
}

        public void play(String title) {
                this.title = title;
                currentTrack = 0;
                System.out.println(description + " playing \"" + title + "\"");
}

        public void play(int track) {
                if (title == null) {
                        System.out.println(description + " can't play track " + currentTrack +
                                        ", no cd inserted");
                } else {
```

```
                currentTrack = track;
                System.out.println(description + " playing track " + currentTrack);
        }
}

public void stop() {
        currentTrack = 0;
        System.out.println(description + " stopped");
}

public void pause() {
        System.out.println(description + " paused \"" + title + "\"");
}

    public String toString() {
        return description;
}
}
```

DvdPlayer class

```
package facade;

public class DvdPlayer {
String description; int
currentTrack; Amplifier
amplifier;String movie;

    public DvdPlayer(String description, Amplifier amplifier) {
        this.description = description;
        this.amplifier = amplifier;
}

public void on() {
        System.out.println(description + " on");
}

public void off() {
        System.out.println(description + " off");
}

   public void eject() {
        movie = null;
      System.out.println(description + " eject");
   }

public void play(String movie) {
```

```
                this.movie = movie;
                currentTrack = 0;
                System.out.println(description + " playing \"" + movie + "\"");
        }

                public void play(int track) {
                        if (movie == null) {
                                System.out.println(description + " can't play track " + track + " no dvd
        inserted");
                        } else {
                        currentTrack = track;
                        System.out.println(description + " playing track " + currentTrack + " of \""
        + movie + "\"");
                        }
        }

        public void stop() {
                        currentTrack = 0;
                        System.out.println(description + " stopped \"" + movie + "\"");
        }

        public void pause() {
                        System.out.println(description + " paused \"" + movie + "\"");
        }

                public void setTwoChannelAudio() {
                        System.out.println(description + " set two channel audio");
        }

                public void setSurroundAudio() {
                        System.out.println(description + " set surround audio");
        }

                public String toString() {
                        return description;
        }
        }
```

PopcornPopper class

```
        package facade;

public class PopcornPopper {String
        description;

                public PopcornPopper(String description) {
                        this.description = description;
        }
```

```
	public void on() {
			System.out.println(description + " on");
	}

	public void off() {
			System.out.println(description + " off");
	}

	public void pop() {
			System.out.println(description + " popping popcorn!");
	}

public String toString() {return
	description;
	   }
	}
```

Projector class

```
	package facade;

	public class Projector {
	String description;
	DvdPlayer dvdPlayer;

		public Projector(String description, DvdPlayer dvdPlayer) {
			this.description = description;
			this.dvdPlayer = dvdPlayer;
	}

	public void on() {
			System.out.println(description + " on");
	}

	public void off() {
			System.out.println(description + " off");
	}

	public void wideScreenMode() {
			System.out.println(description + " in widescreen mode (16x9 aspect ratio)");
	}

	public void tvMode() {
			System.out.println(description + " in tv mode (4x3 aspect ratio)");
	}

		public String toString() {
```

```
            return description;
        }
    }
```

Screen class

```
package facade;

public class Screen {
String description;

        public Screen(String description) {
            this.description = description;
}

public void up() {
            System.out.println(description + " going up");
}

public void down() {
            System.out.println(description + " going down");
}


        public String toString() {
            return description;
}
}
```

TheaterLights class

```
package facade;

public class TheaterLights {String
    description;

        public TheaterLights(String description) {
            this.description = description;
    }

    public void on() {
            System.out.println(description + " on");
    }

    public void off() {
            System.out.println(description + " off");
    }

    public void dim(int level) {
            System.out.println(description + " dimming to " + level  + "%");
```

```
}

        public String toString() {
                return description;
}
}
```

Tuner class

```
package facade;

public class Tuner {
String description;
Amplifier amplifier;double
frequency;

        public Tuner(String description, Amplifier amplifier) {
                this.description = description;
}

public void on() {
                System.out.println(description + " on");
}

public void off() {
                System.out.println(description + " off");
}

public void setFrequency(double frequency) {
                System.out.println(description + " setting frequency to " + frequency);
                this.frequency = frequency;
}

public void setAm() {
                System.out.println(description + " setting AM mode");
}

public void setFm() {
                System.out.println(description + " setting FM mode");
}

        public String toString() {
                return description;
}
}
```

HomeTheaterTestDrive class

```
package facade;
```

```
public class HomeTheaterTestDrive {
public static void main(String[] args) {
        Amplifier amp = new Amplifier("Top-O-Line Amplifier");
        Tuner tuner = new Tuner("Top-O-Line AM/FM Tuner", amp);
        DvdPlayer dvd = new DvdPlayer("Top-O-Line DVD Player", amp);
        CdPlayer cd = new CdPlayer("Top-O-Line CD Player", amp);
        Projector projector = new Projector("Top-O-Line Projector", dvd);
        TheaterLights lights = new TheaterLights("Theater Ceiling Lights");
        Screen screen = new Screen("Theater Screen");
        PopcornPopper popper = new PopcornPopper("Popcorn Popper");

        HomeTheaterFacade homeTheater =
                new HomeTheaterFacade(amp, tuner, dvd, cd,
                        projector, screen, lights, popper);

        homeTheater.watchMovie("Raiders of the Lost Ark");
        homeTheater.endMovie();
}
}
```

Output:

```
Get ready to watch a movie...
Popcorn Popper on
Popcorn Popper popping popcorn!
Theater Ceiling Lights dimming to 10%
Theater Screen going down
Top-O-Line Projector on
Top-O-Line Projector in widescreen mode (16x9 aspect ratio)
Top-O-Line Amplifier on
Top-O-Line Amplifier setting DVD player to Top-O-Line DVD Player
Top-O-Line Amplifier surround sound on (5 speakers, 1 subwoofer)
Top-O-Line Amplifier setting volume to 5
Top-O-Line DVD Player on
Top-O-Line DVD Player playing "Raiders of the Lost Ark"
Shutting movie theater down...
Popcorn Popper off
Theater Ceiling Lights on
Theater Screen going up
Top-O-Line Projector off
Top-O-Line Amplifier off
Top-O-Line DVD Player stopped "Raiders of the Lost Ark"
Top-O-Line DVD Player eject
Top-O-Line DVD Player off
```

Q.2) Write a python program to make Categorical values in numeric format for a given dataset

```
#import pandas
import pandas as pd

# read csv file
df = pd.read_csv('Customers.csv')
print(df)
print("\n After repalcing Category Male as 0 and Female as 1");
# replacing values

df['GENDER'].replace(['Male', 'Female'],
                        [0, 1], inplace=True)
print(df)
```

Output:

```
Name  Episodes  Gender
0 Sheldon   42  male
1  Penny   24  female
2   Amy    31 female
3  Penny   29 female
4   Raj   37  male
5 Sheldon   40  male
   Name  Episodes  Gender  female  male
0 Sheldon   42  male   0  1
1  Penny   24  female  1  0
2   Amy    31  female  1  0
3  Penny   29  female  1  0
4   Raj   37  male   0  1
5 Sheldon   40  male   0  1
```

Q.3) Write node js script to build Your Own Node.js Module. Use require ('http') module is a built-in Node module that invokes the functionality of the HTTP library to create a local server. Also use the export statement to make functions in your module available externally. Create a new text file to contain the functions in your module called, "modules.js" and add this function to return today's date and time.

Firstmodule.js

```
exports.myDateTime = function () {
    return Date();
  };
```

Demo_module.js

```
var http = require('http');
var dt = require('./firstmodule');
```

```
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write("The date and time are currently: " +
  dt.myDateTime());res.end();
}).listen(8080);
```

Output:

The date and time are currently: Sat Feb 12 2022 12:49:07 GMT+0530 (India Standard Time)

Q.1) Write a Java Program to implement Observer Design Pattern for number conversion. Accept a number in Decimal form and represent it in Hexadecimal, Octal and Binary. Changethe Number and it reflects in other forms also

Subject Interface

```
public interface Subject {
 /*
  * Both of these methods take an Observer as an argument; that is, the
  * Observer to be registered or removed
  */
 public void registerObserver(Observer o);

 public void removeObserver(Observer o);

 /*
  * This method is called to notify all observers when the Subject's state
  * has changed.
  */
 public void notifyObserver();
}
```

Observer Interface

```
public interface Observer {
 /*
  * The Observer interface is implemented by all observers, so they all have
  * to implement the update() method.
  */
 public void update();
}
```

DisplayElement Interface

```
public interface DisplayElement {
 /*
  * The DisplayElement interface just includes one method, display(), that we
  * will call when the display element needs to be displayed.
  */
 public void display();
}
```

DecimalData Class

```
import java.util.ArrayList;
import java.util.List;
```

```
public class DecimalData implements Subject {
  // ArrayList to hold the Observers, and we create it in the constructor.private
  List<Observer> observers;
  private int decimalValue;

  public DecimalData() {
    observers = new ArrayList<Observer>();
  }
```

```
// When an observer registers, we just add it to the end of the list.@Override
public void registerObserver(Observer o) {
  observers.add(o);
}

// when an observer wants to unregister, we just take it off the list.@Override
public void removeObserver(Observer o) {int
  index = observers.indexOf(o);
  if (index >= 0) {
    observers.remove(index);
  }
}

/*
  * tell all the observers about the state. Because they are all Observers,
  * we know they all implement update(), so we know how to notify them.
  */
@Overrid
e
public void notifyObserver() {
  for (Observer observer : observers) {
    observer.update();
  }
}

// notify the Observers when decimal value change , means sate of object
// changed.
public void valuesChanged() {
  notifyObserver();
}

public int getDecimalValue() {
  return decimalValue;
}

// we're going to use this method to test our display elementspublic void
setDecimalValue(int decimalValue) { this.decimalValue = decimalValue;
  valuesChanged();
  }
}
```

BinaryObserver Class

```
public class BinaryObserver implements Observer, DisplayElement {
  private Subject decimalSubject;
  private int decimalValue;

  public BinaryObserver(Subject decimalSubject) {
    this.decimalSubject = decimalSubject;
    decimalSubject.registerObserver(this);
  }

  @Override
  public void update() { DecimalData
    decimalData = null;
    if (decimalSubject instanceof DecimalData) { decimalData =
      (DecimalData) decimalSubject; decimalValue =
      decimalData.getDecimalValue();
    }
    display();
  }

  /*
   * The display() method just prints out the most recent decimal value in
   * binary.
   */
  @Override
  public void display() { System.out.println("Binary String: "
        + Integer.toBinaryString(decimalValue));
  }
}
```

OctalObserver Class

```
public class OctalObserver implements Observer, DisplayElement {
  private Subject decimalSubject;
  private int decimalValue;

  public OctalObserver(Subject decimalSubject) {
    this.decimalSubject = decimalSubject;
    decimalSubject.registerObserver(this);
  }

  @Override
  public void update() { DecimalData
    decimalData = null;
    if (decimalSubject instanceof DecimalData) { decimalData =
      (DecimalData) decimalSubject; decimalValue =
      decimalData.getDecimalValue();
    }
    display();
  }

  /*
   * The display() method just prints out the most recent decimal value in
   * octal.
   */
  @Override
  public void display() {
```

```
    System.out.println("Octal String: "
        + Integer.toOctalString(decimalValue));
  }
}
```

HexObserver Class

```
public class HexObserver implements Observer, DisplayElement {
  private Subject decimalSubject;
  private int decimalValue;

  public HexObserver(Subject decimalSubject) {
    this.decimalSubject = decimalSubject;
    decimalSubject.registerObserver(this);
  }

  @Override
  public void update() { DecimalData
    decimalData = null;
    if (decimalSubject instanceof DecimalData) { decimalData =
      (DecimalData) decimalSubject; decimalValue =
      decimalData.getDecimalValue();
    }
    display();
  }

  /*
   * The display() method just prints out the most recent decimal value in
   * hexadecimal.
   */
  @Override
  public void display() {
    System.out.println("Hex String: " + Integer.toHexString(decimalValue));
  }
}
```

ObserverPatternDemo

```
public class ObserverPatternDemo {

  public static void main(String[] args) {
    // First, create the DecimalData object. DecimalData decimalData =
    new DecimalData();

    /*
     * Create the three format Object and pass them the DecimalData object.
     */

    BinaryObserver binaryObserver = new BinaryObserver(decimalData);OctalObserver
    octalObserver = new OctalObserver(decimalData); HexObserver hexObserver = new
    HexObserver(decimalData);
    // set Decimal Value decimalData.setDecimalValue(24);
    System.out.println("\nNow, Data is changed.\n");
    // Now, new decimal value
    decimalData.setDecimalValue(124);
  }
}
```

Output-

```
Binary String: 11000
Octal String: 30
Hex String: 18

Now, Data is changed.

Binary String: 1111100
Octal String: 174
Hex String: 7c
```

Q.2) Write a python program to Implement Simple Linear Regression for predicting houseprice.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pltimport
seaborn as sns

#sns.set_style("whitegrid")
#plt.style.use("fivethirtyeight")

USAhousing = pd.read_csv('USA_Housing.csv')
USAhousing.head()

X = USAhousing[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number ofRooms',
                'Avg. Area Number of Bedrooms', 'Area Population']]y =
USAhousing['Price']

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,random_state=101)


from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression(normalize=True)
lin_reg.fit(X_train,y_train)

pred = lin_reg.predict(X_test)
plt.scatter(y_test, pred) plt.show()
```

Q.3) Create a js file named main.js for event-driven application. There should be a main loopthat listens for events, and then triggers a callback function when one of those events is detected.

```
// Import events module
```

```
var events = require('events');
// Create an eventEmitter object
var eventEmitter = new events.EventEmitter();

// Create an event handler as follows
var connectHandler = function connected() {
    console.log('connection succesful.');

    // Fire the data_received event
    eventEmitter.emit('data_received');
}

// Bind the connection event with the handler eventEmitter.on('connection',
connectHandler);
 // Bind the data_received event with the anonymous functioneventEmitter.on('data_received',
function(){
    console.log('data received succesfully.');
});
// Fire the connection event
eventEmitter.emit('connection');
console.log("Program Ended.");
```

Output-
connection succesful.
data received succesfully.
Program Ended.

Q.1) Write a java program to implement abstract factory pattern for shape interface.

1) Interface

Shape.java

```
package javaprograms;
public interface Shape {
        void draw();
}
```

Color.java

```
package javaprograms;
public interface Color {
        void fill();
}
```

2) Create Class

Rectangle.java package
javaprograms;

```
public class Rectangle implements Shape {@Override
        public void draw() {
        System.out.println("Inside Rectangle::draw() method.");
        }
}
```

Sqaure.java

```
package javaprograms;

public class Square implements Shape {
        @Override
        public void draw() {
        System.out.println("Inside Square::draw() method.");
        }
}
```

Circle.java

```
package javaprograms;
public class Circle implements Shape {@Override
        public void draw() {
        System.out.println("Inside Circle::draw() method.");
        }
}
```

Red.java

```
package javaprograms;
public class Red implements Color {
        @Override
        public void fill() {
        System.out.println("Inside Red::fill() method.");
        }
}
```

Green.java

```
package javaprograms;

public class Green implements Color {
        @Override
        public void fill() {
        System.out.println("Inside Green::fill() method.");
        }
}
```

Blue.java

```
package javaprograms;

public class Blue implements Color {
        @Override
        public void fill() {
        System.out.println("Inside Blue::fill() method.");
        }
}
```

AbstractFactory.java

```
package javaprograms;

public abstract class AbstractFactory { abstract Color
        getColor(String color); abstract Shape
        getShape(String shape) ;
       }
```

ShapeFactory.java

```
package javaprograms;
public class ShapeFactory extends AbstractFactory {@Override
        public Shape getShape(String shapeType){

        if(shapeType == null){return
        null;
        }
```

```
        if(shapeType.equalsIgnoreCase("CIRCLE")){return
        new Circle();

        }else if(shapeType.equalsIgnoreCase("RECTANGLE")){return
        new Rectangle();

        }else if(shapeType.equalsIgnoreCase("SQUARE")){return
        new Square();
        }

        return null;
        }

        @Override
        Color getColor(String color) {
        return null;
        }
}
```

ColorFactory.java

```
package javaprograms;

public class ColorFactory extends AbstractFactory {@Override
        public Shape getShape(String shapeType){
        return null;
        }

        @Override
        Color getColor(String color) {

        if(color == null){return
        null;
        }

        if(color.equalsIgnoreCase("RED")){return
        new Red();

        }else if(color.equalsIgnoreCase("GREEN")){return new
        Green();

        }else if(color.equalsIgnoreCase("BLUE")){return new
        Blue();
        }
```

```
        return null;
        }
      }
```

FactoryProducer.java

```
package javaprograms;

public class FactoryProducer {
        public static AbstractFactory getFactory(String choice){

        if(choice.equalsIgnoreCase("SHAPE")){return
        new ShapeFactory();

        }else if(choice.equalsIgnoreCase("COLOR")){return new
        ColorFactory();
        }

        return null;
        }
      }
```

AbstractFactoryPatternDemo.java

```
package javaprograms;

public class AbstractFactoryPatternDemo {
        public static void main(String[] args) {
        //get shape factory AbstractFactory
        shapeFactory =
FactoryProducer.getFactory("SHAPE");
        //get an object of Shape Circle
        Shape shape1 = shapeFactory.getShape("CIRCLE");
        //call draw method of Shape Circle
        shape1.draw();
        //get an object of Shape Rectangle
        Shape shape2 = shapeFactory.getShape("RECTANGLE");
        //call draw method of Shape Rectangle
        shape2.draw();

        //get an object of Shape Square
        Shape shape3 = shapeFactory.getShape("SQUARE");
        //call draw method of Shape Square
        shape3.draw();
        //get color factory AbstractFactory
        colorFactory =
FactoryProducer.getFactory("COLOR");
```

```
        //get an object of Color Red
        Color color1 = colorFactory.getColor("RED");
        //call fill method of Red
        color1.fill();
        //get an object of Color Green
        Color color2 = colorFactory.getColor("Green");
        //call fill method of Green
        color2.fill();
        //get an object of Color Blue
        Color color3 = colorFactory.getColor("BLUE");
        //call fill method of Color Bluecolor3.fill();
        }

}
```

Output-

```
Inside Circle::draw() method. Inside
Rectangle::draw() method.Inside
Square::draw() method.
Inside Red::fill() method. Inside
Green::fill() method.Inside Blue::fill()
method.
```

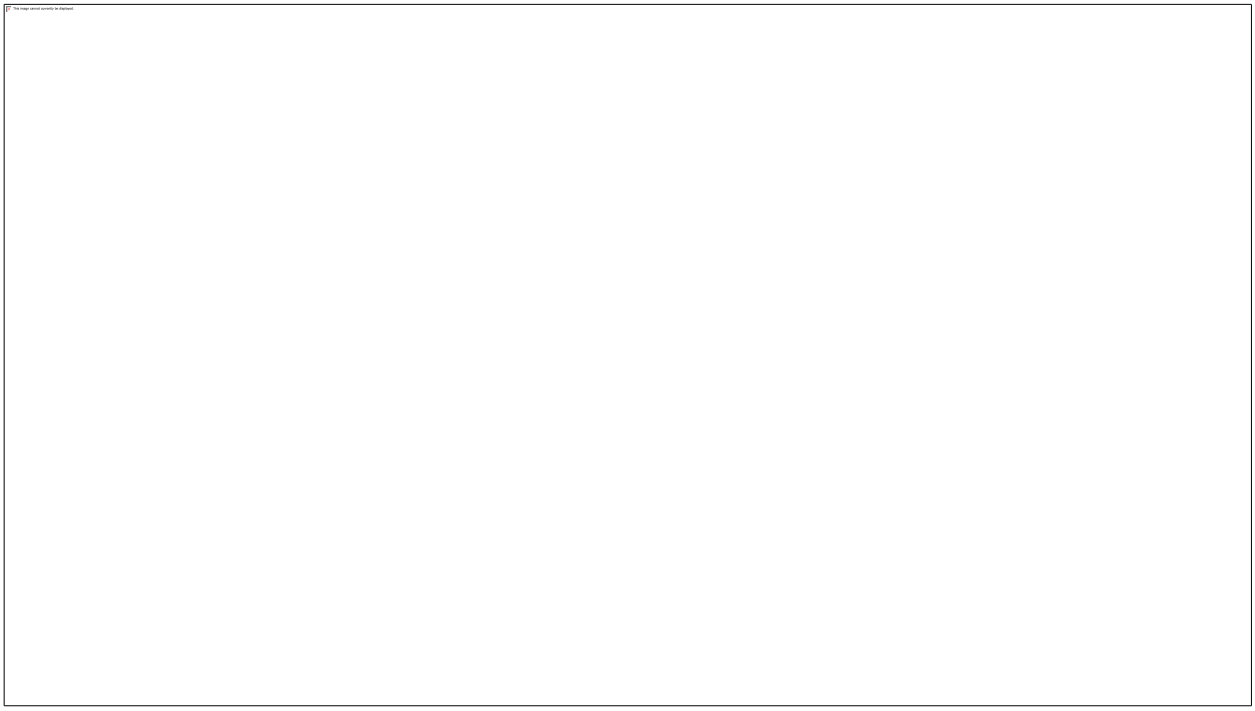
Q.2) Write a python program to implement Multiple Linear Regression for a given dataset.

```
import matplotlib.pyplot as pltimport
pandas as pd
import numpy as np
import pylab as pl


df = pd.read_csv('Fuelconsumption.csv')df.head()

cdf =
df[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_CITY','FUELCONSUMPTION_HWY','FUELCO
NSUMPTION_COMB','CO2EMISSIONS']]
cdf.head()

plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS, color='blue')
plt.xlabel('Engine Size')
plt.ylabel('Emissions')plt.show()

msk = np.random.rand(len(df)) < 0.8train =
cdf[msk]
test = cdf[~msk]
```

```
from sklearn import linear_model
regr = linear_model.LinearRegression()
x = np.asanyarray(train[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
y = np.asanyarray(train[['CO2EMISSIONS']])regr.fit(x,y)

print('Coefficients: ', regr.coef_)
```

Q.3) Write node js application that transfer a file as an attachment on web and enables browser to prompt the user to download file using express js.

```
var express = require('express');var app =
express();
var PORT = 3000;
app.get('/', function(req, res){ res.download('message.txt', function(error){
                console.log("Error : ", error)
        });
});
app.listen(PORT, function(err){if (err)
        console.log(err);
        console.log("Server listening on PORT", PORT);
});
```



Q.1 Write a Java Program to implement Factory method for Pizza Store with createPizza(), orederPizza(), prepare(), Bake(), cut(), box(). Use this to create variety of pizza's like NyStyleCheesePizza, ChicagoStyleCheesePizza etc

Create Class –
1)Pizza.class

```
package javaprograms;
import java.util.ArrayList;

abstract public class Pizza {String
	name;
	String dough;
	String sauce;
	ArrayList toppings = new ArrayList();

	public String getName() {
		return name;
	}

	public void prepare() { System.out.println("Preparing " +
		name);
	}

	public void bake() { System.out.println("Baking " +
		name);
	}

	public void cut() { System.out.println("Cutting " +
		name);
	}

	public void box() { System.out.println("Boxing " +
		name);
	}

	public String toString() {
		// code to display pizza name and ingredientsStringBuffer
		display = new StringBuffer(); display.append("---- " + name
		+ " ---------------------------------------------------- \n");
		display.append(dough + "\n");
		display.append(sauce + "\n");
		for (int i = 0; i < toppings.size(); i++) { display.append((String) toppings.get(i) +
			"\n");
		}
		return display.toString();
	}
}
```

2)PizzaStore.class

```
package javaprograms;

public class PizzaStore {
	SimplePizzaFactory factory;

	public PizzaStore(SimplePizzaFactory factory) {
		this.factory = factory;
	}

	public Pizza orderPizza(String type) {Pizza pizza;

		pizza = factory.createPizza(type);

		pizza.prepare();
		pizza.bake();
		pizza.cut();
		pizza.box();

		return pizza;
	}
}
```

3)SimplePizzaFactory.class

```
package javaprograms;

public class SimplePizzaFactory {

	public Pizza createPizza(String type) {Pizza pizza =
		null;

		if (type.equals("cheese")) { pizza = new
			NYCheesePizza();
		} else if (type.equals("veggie")) { pizza = new
			ChicagoCheesePizza();
		}
		return pizza;
	}
}
```

4)NYCheesePizza.class

```
package javaprograms;
public class NYCheesePizza extends Pizza {
	public   NYCheesePizza()   { name =
		"NY Cheese Pizza";dough =
		"Regular Crust";
		sauce = "Marinara Pizza Sauce";
```

```
            toppings.add("Fresh Mozzarella");
            toppings.add("Parmesan");
        }
}
```

5)ChicagoCheesePizza.class

```
package javaprograms;
public class ChicagoCheesePizza extends Pizza {
        public   ChicagoCheesePizza()   { name =
            "Chicago Cheese Pizza";dough =
            "Crust";
            sauce = "Marinara sauce";
            toppings.add("Shredded mozzarella");
            toppings.add("Grated parmesan");
            toppings.add("Diced onion");
            toppings.add("Sliced mushrooms");
            toppings.add("Sliced red pepper");
            toppings.add("Sliced black olives");
        }
}
```

Output-

Preparing NY Cheese Pizza
Baking NY Cheese Pizza Cutting
NY Cheese Pizza Boxing NY
Cheese Pizza
We ordered a NY Cheese Pizza

Preparing Chicago Cheese PizzaBaking
Chicago Cheese Pizza Cutting Chicago
Cheese Pizza Boxing Chicago Cheese
Pizza
We ordered a Chicago Cheese Pizza

Q.2. Write a python program to implement Naive Bayes.

```
from sklearn import datasetsfrom
sklearn import metrics
from sklearn.naive_bayes import GaussianNB



dataset = datasets.load_iris()

#Creating our Naive Bayes Modelmodel =
GaussianNB()
```

```
model.fit(dataset.data, dataset.target)

#Making Predictions expected =
dataset.target
predicted = model.predict(dataset.data)

#Getting Accuracy and Statistics print(metrics.classification_report(expected,
predicted))print(metrics.confusion_matrix(expected, predicted))
```

Q.3 Design a Django application that adds web pages with views and templates.

home.html

```
<h2>Hello {{name}}</h2>
<form action="add/">
    Enter 1st number<input type="text" name="num1"><br>Enter 2nd
    number<input type="text" name="num2"><br>
    <input type="submit">
</form>
```

result.html

```
result:{{result}}
```

views.py

```
from django.http import HttpResponse def
name(request):
    return render(request,"home.html",{"name":"Akshada"}) def
add(request):
    val1=int(request.GET['num1'])
    val2=int(request.GET['num2'])
    res=val1+val2
    return render(request,"result.html",{'result':res})
```

urls.py

```
from django.urls import pathfrom
.views import add,name urlpatterns =
[
    path("", name,name="home"),
    path('add/',add,name="home"),

]
```

test_project

```
from django.contrib import adminfrom
django.urls import path from django.urls
import include urlpatterns = [
```

```
    path('home/', include("home.urls")),
]
```

Q.1 Write a Java Program to implement I/O Decorator for converting uppercase letters to lower case letters.

1. Create Interface:

```
public interface ToLowerDecorator {
        public void lower(String ch);

}
```

2. LowerCase.java

```
package javaprograms;
import java.lang.*;
import java.io.*;
public class LowerCase implements ToLowerDecorator{
        public void lower(String ch)

        {
                ch=ch.toLowerCase();
                System.out.println("Lowercase:"+ch);

        }
}
```

3.Decorator.java

```
package javaprograms;import java.io.*;
import java.util.Scanner;
public class Decorator {
        public static void main(String[] args) {

                // TODO Auto-generated method stub
                ToLowerDecorator l=new LowerCase();
                //l.lower("HeLLO");

                Scanner sc=new Scanner(System.in);
            System.out.println("enter character:"); String
                        s=sc.nextLine();
            System.out.println("entered character:"+s);
              l.lower(s);

        }
}
```

Output –

```
enter character:
HELLO
```

entered character:HELLO
Lowercase:hello

Q.2. Write a python program to implement Decision Tree whether or not to play Tennis.

```
# Write a python program to Implement Decision Tree whether or not to play tennis.
import numpy as np
```

```
import pandas as pd
import matplotlib.pyplot as plt

PlayTennis = pd.read_csv("PlayTennis.csv")

#We can convert all the non numerical values into numerical values using
LabelEncoder

from sklearn.preprocessing import LabelEncoderLe =
LabelEncoder()

PlayTennis['outlook'] = Le.fit_transform(PlayTennis['outlook']) PlayTennis['temp'] =
Le.fit_transform(PlayTennis['temp']) PlayTennis['humidity'] =
Le.fit_transform(PlayTennis['humidity'])PlayTennis['windy'] =
Le.fit_transform(PlayTennis['windy']) PlayTennis['play'] =
Le.fit_transform(PlayTennis['play'])

#Lets split the training data and its coresponding prediction values.#y - holds all
the decisions.
#X - holds the training data.y =
PlayTennis['play']
X = PlayTennis.drop(['play'],axis=1)

# Fitting the model
from sklearn import tree
clf = tree.DecisionTreeClassifier(criterion = 'entropy')   #A decision treeclassifier. “entropy” for
the information gain.
clf = clf.fit(X, y)                                                    #Decision tree
algorithm splits nodes as long as this value decreases till it reaches zero

# We can visualize the tree using tree.plot_tree
tree.plot_tree(clf)
plt.show()
```

Q.3 Develop a basic poll application (app).It should consist of two parts:

a) A public site in which user can pick their favourite programming language and vote.

b) An admin site that lets you add, change and delete programming languages.a)Public site

Languages.html

```
<form action="select/">
    <label>Select Language</label>
    <!-- <input type="radio" id="r1" name="r1" value="JAVA">JAVA<br>
    <input type="radio" id="r1" name="r1" value="PHP">PHP<br>
    <input type="radio" id="r1" name="r1" value="Javascript">Javascript<br> -->
```

{% for d in lang %}

```
        <input type="radio" id="r1" name="r1"
value="{{d.language}}">{{d.language}}<br>
        {%endfor%}
 <!--
  -->
        <input type="submit" name='vote' value="Vote">

        <table>
            <thead>
                <tr>
                <th>Language</th>
                <th>Total Voting</th>
            </tr>
            </thead>
            <tbody>
                {% for c in res %}
                <tr>
                    <td>{{c.langname}}</td>
                    <td>{{c.name_count}}</td>
                </tr>
                {%endfor%}
            </tbody>
        </table>
</form>
```

models.py

```
from django.db import models
from django.contrib.auth.models import User


# Create your models here. class
Language(models.Model):
        langname=models.CharField(max_length=63)

STATUS = (
        (0,"Draft"),
        (1,"Publish")
)
class AdminLang(models.Model):
        language=models.CharField(max_length=63)
        status = models.IntegerField(choices=STATUS, default=0)class
        Meta:
            verbose_name_plural = "AdminLang"
```

```
        def__str__(self): return
            self.language
```

views.py

```
from django.shortcuts import render
from language.models import Language,AdminLang
from django.db.models import Count

# Create your views here.
from django.http import HttpResponse

def language(request):
            data=AdminLang.objects.all()data1={"lang":data}
            return render(request,"displaylang.html",data1)

def select(request): langname=request.GET.get('r1')
        result=Language.objects.create(langname=langname).save()
        count=Language.objects.values('langname').annotate(name_count=Count('langname
')).filter(name_count__gt=1)
        return render(request,"displaylang.html",{"res":count})
```

urls.py

```
from django.urls import path
from .views import language,select

urlpatterns = [ path("lang/",language,name="language"),
        path("lang/select/",select,name="language"),
]
```

test_project

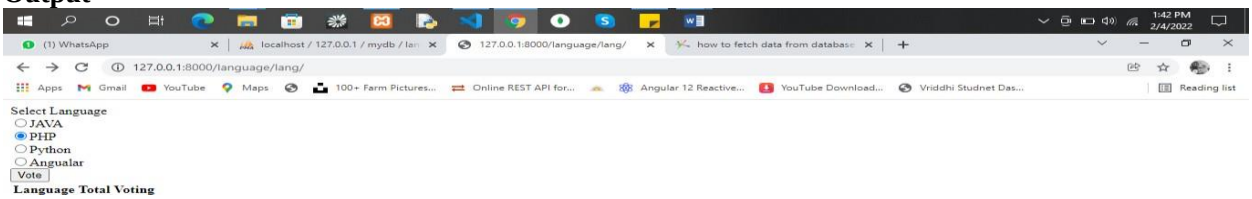
urls.py

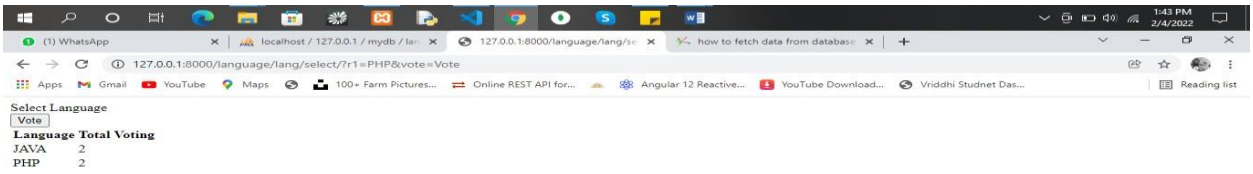
```
from django.contrib import admin from
django.urls import path,include

urlpatterns = [
        path('admin/', admin.site.urls),
        path('language/',include("language.urls")),
]
```

b)Admin site

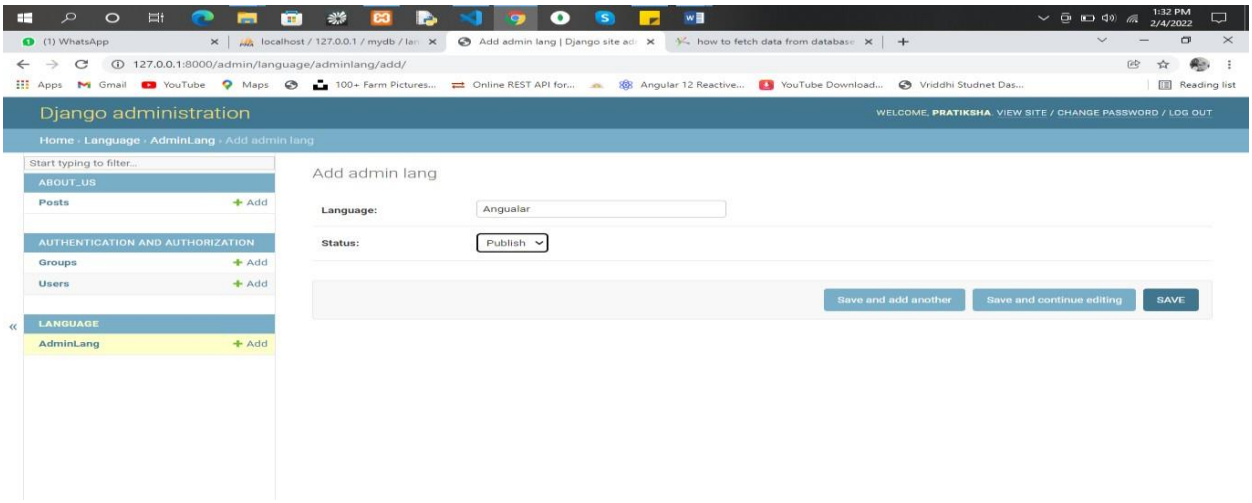
```
from django.contrib import admin

from language.models import AdminLang

@admin.register(AdminLang)
class AdminLangAdmin(admin.ModelAdmin):
        pass
```

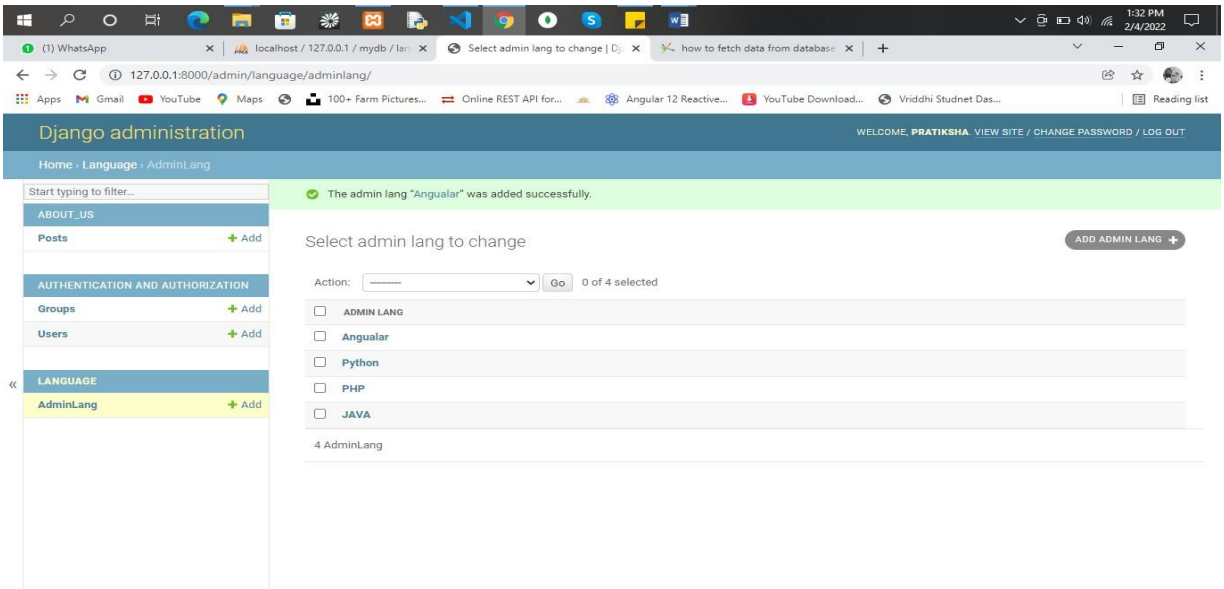
Output –





Admin site:





Q.1 Write a Java Program to implement command pattern to test Remote Control .

1) Create Interface- Command.java

```
package javaprograms;

interface Command
{
	public void execute();
}
```

2) Create Class Light.java

```
package javaprograms;
public class Light
{
	public void on()
	{
		System.out.println("Light is on");
	}
	public void off()
```

```
        {
                System.out.println("Light is off");
        }
}
```

LightOnCommand.java

```
package javaprograms;
class LightOnCommand implements Command
{
        Light light;

        // The constructor is passed the light it
        // is going to control.
        public LightOnCommand(Light light)
        {
            this.light = light;
        }
        public void execute()
        {
            light.on();
        }
}
```

LightOffCommand.java

```
package javaprograms;
class LightOffCommand implements Command
{
        Light light;
        public LightOffCommand(Light light)
        {
             this.light = light;
        }
        public void execute()
        {
```

```
            light.off();
        }
    }
```

Stereo.java

```
package javaprograms;

public class Stereo
{
public void on()
{
            System.out.println("Stereo            is on");
}
public void off()
{
            System.out.println("Stereo            is off");
}
public void setCD()
{
                    System.out.println("Stereo  is set " +
                                        "for CD  input");
}
public void setDVD()
{
                    System.out.println("Stereo  is set"+
                                      " for DVD  input");
}
public void setRadio()
{
                    System.out.println("Stereo  is set" +
                                       " for Radio");
        }
        public void setVolume(int volume)
        {
            // code to set the volume System.out.println("Stereo volume
            set"
                                       + " to " + volume);
        }
}
```

StereoOffCommand.java

```
package javaprograms;

class StereoOffCommand implements Command
{
        Stereo stereo;
        public StereoOffCommand(Stereo stereo)
        {
            this.stereo = stereo;
        }
        public void execute()
        {
            stereo.off();
```

```
        }
}
```

StereoOnWithCDCommand.java

```
package javaprograms;

class StereoOnWithCDCommand implements Command
{
        Stereo stereo;
        public StereoOnWithCDCommand(Stereo stereo)
        {
                this.stereo = stereo;
        }
        public void execute()
        {
                stereo.on(); stereo.setCD();
                stereo.setVolume(11);
        }
}
```

SimpleRemoteControl.java

```
package   javaprograms; class
SimpleRemoteControl
{
        Command slot;   // only one button

        public SimpleRemoteControl()
        {
        }

        public void setCommand(Command command)
        {
                // set the command the remote will
                //   execute slot =
                command;
        }
        public void buttonWasPressed()
        {
                slot.execute();
        }
}
```

RemoteControlTest.java

```
package javaprograms;

class RemoteControlTest
{
        public static void main(String[] args)
        {
                SimpleRemoteControl remote =
                                new SimpleRemoteControl();Light
                light = new Light();
                Stereo stereo = new Stereo();
```

```
        // we can change command dynamically
        remote.setCommand(new
                        LightOnCommand(light));
        remote.buttonWasPressed(); remote.setCommand(new
                StereoOnWithCDCommand(stereo));
        remote.buttonWasPressed(); remote.setCommand(new
                        StereoOffCommand(stereo));
        remote.buttonWasPressed();
    }
}
```

Output- Light is
on Stereo is on
Stereo is set for CD inputStereo
volume set to 11 Stereo is off

Q.2. Write a python program to implement Linear SVM.

```
# Import the Libraries
import numpy as np
import matplotlib.pyplot as plt from
sklearn import svm, datasets

# Import some Data from the iris Data Setiris =
datasets.load_iris()

# Take only the first two features of Data.
# To avoid the slicing, Two-Dim Dataset can be used

X = iris.data[:, :2]y =
iris.target

# C is the SVM regularization parameterC =
1.0

# Create an Instance of SVM and Fit out the data.
# Data is not scaled so as to be able to plot the support vectors
svc = svm.SVC(kernel ='linear', C = 1).fit(X, y) #Fit the SVM model according tothe given
training data.
                                                    #SVC=Support Vector Classifier

# create a mesh to plot
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
```

```
h = (x_max / x_min)/100
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),                #Return coordinate matrices
from coordinate vectors.
            np.arange(y_min, y_max, h))

# Plot the data for Proper Visual Representation
plt.subplot(1, 1, 1) #Add a subplot to the current figure.subplot(nrows, ncols,indexOfSubplot)

# Predict the result by giving Data to the model
Z = svc.predict(np.c_[xx.ravel(), yy.ravel()])                #ravel()-Return a contiguous
flattened array.
Z = Z.reshape(xx.shape)
plt.contourf(xx, yy, Z, cmap = plt.cm.Paired, alpha = 0.8) #contour and contourfdraw contour lines
and filled contours.

plt.scatter(X[:, 0], X[:, 1], c = y, cmap = plt.cm.Paired)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width') plt.xlim(xx.min(),
xx.max()) plt.title('SVC with linear kernel')

# Output the Plot
plt.show()
```

Q.3 Design a Django application: A public site in which user can pick their favouriteprogramming language and vote.

Language.html

```
<form action="select/">
      <label>Select Language</label><br>
      <!-- <input type="radio" id="r1" name="r1" value="JAVA">JAVA<br>
      <input type="radio" id="r1" name="r1" value="PHP">PHP<br>
      <input type="radio" id="r1" name="r1" value="Javascript">Javascript<br> -->
      {% for d in lang %}
      <input type="radio" id="r1" name="r1"
value="{{d.language}}">{{d.language}}<br>
      {%endfor%}

      <input type="submit" name='vote' value="Vote">

      <table border="1">
            <thead>
                  <tr>
                  <th>Language</th>
                  <th>Total Voting</th>
```

```
            </tr>
            </thead>
            <tbody>
                {% for c in res %}
                <tr>
                    <td>{{c.langname}}</td>
                    <td>{{c.name_count}}</td>
                </tr>
                {%endfor%}
            </tbody>
        </table>
</form>
```

models.py

```
from django.db import models
from django.contrib.auth.models import User


# Create your models here. class
Language(models.Model):
    langname=models.CharField(max_length=63)

STATUS = (
    (0,"Draft"),
    (1,"Publish")
)
class AdminLang(models.Model):
    language=models.CharField(max_length=63)
    status = models.IntegerField(choices=STATUS, default=0)class
    Meta:
        verbose_name_plural = "AdminLang"

    def__str__(self): return
        self.language
```

views.py

```
from django.shortcuts import render
from language.models import Language,AdminLangfrom
django.db.models import Count

# Create your views here.
from django.http import HttpResponsedef
```

```
language(request):
```

```
        data=AdminLang.objects.all()
        data1={"lang":data}
        return render(request,"displaylang.html",data1)

def select(request): langname=request.GET.get('r1')
    result=Language.objects.create(langname=langname).save()
    count=Language.objects.values('langname').annotate(name_count=Count('langname
')).filter(name_count__gt=1)
    return render(request,"displaylang.html",{"res":count})
```

urls.py

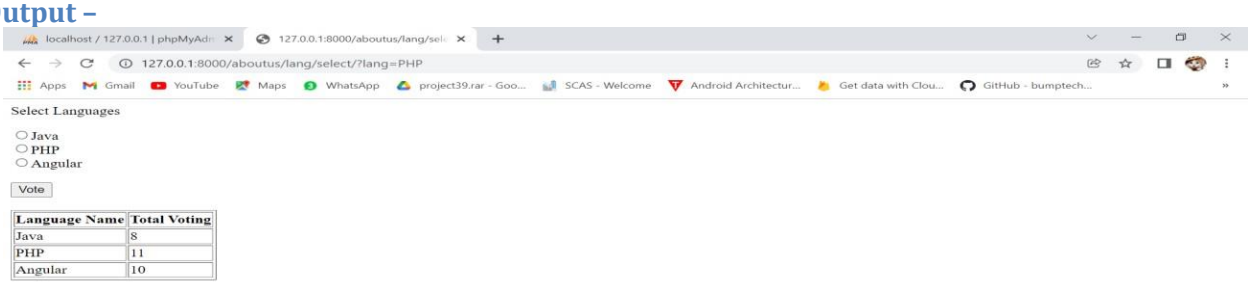
```
from django.urls import path
from .views import language,select

urlpatterns = [ path("lang/",language,name="language"),
    path("lang/select/",select,name="language"),
]
```

test_project

urls.py

```
from django.contrib import admin from
django.urls import path,include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('language/',include("language.urls")),
]
```

Output –



Q.1 Design simple HR Application using Spring Framework

Q.2 Write a Python program to prepare Scatter Plot for Iris Dataset

```
import pandas as pd
import matplotlib.pyplot as plt
iris=pd.read_csv("Iris.csv")
iris.plot(kind="scatter",x='SepalLengthCm',y='PetalLengthCm')plt.grid()
plt.show()
```

Q.3 Design a Django application: An admin site that lets you add, change and delete programming languages

Views.py

```
from django.shortcuts import render
from language.models import Language,AdminLangfrom
django.db.models import Count

# Create your views here.
from django.http import HttpResponse

def language(request):
        data=AdminLang.objects.all()
        data1={"lang":data}
        return render(request,"displaylang.html",data1)

def select(request): langname=request.GET.get('r1')
    result=Language.objects.create(langname=langname).save()
    count=Language.objects.values('langname').annotate(name_count=Count('langname
')).filter(name_count__gt=1)
    return render(request,"displaylang.html",{"res":count})
```

urls.py

```
from django.urls import path
from .views import language,select

urlpatterns = [ path("lang/",language,name="language"),
    path("lang/select/",select,name="language"),
]
```

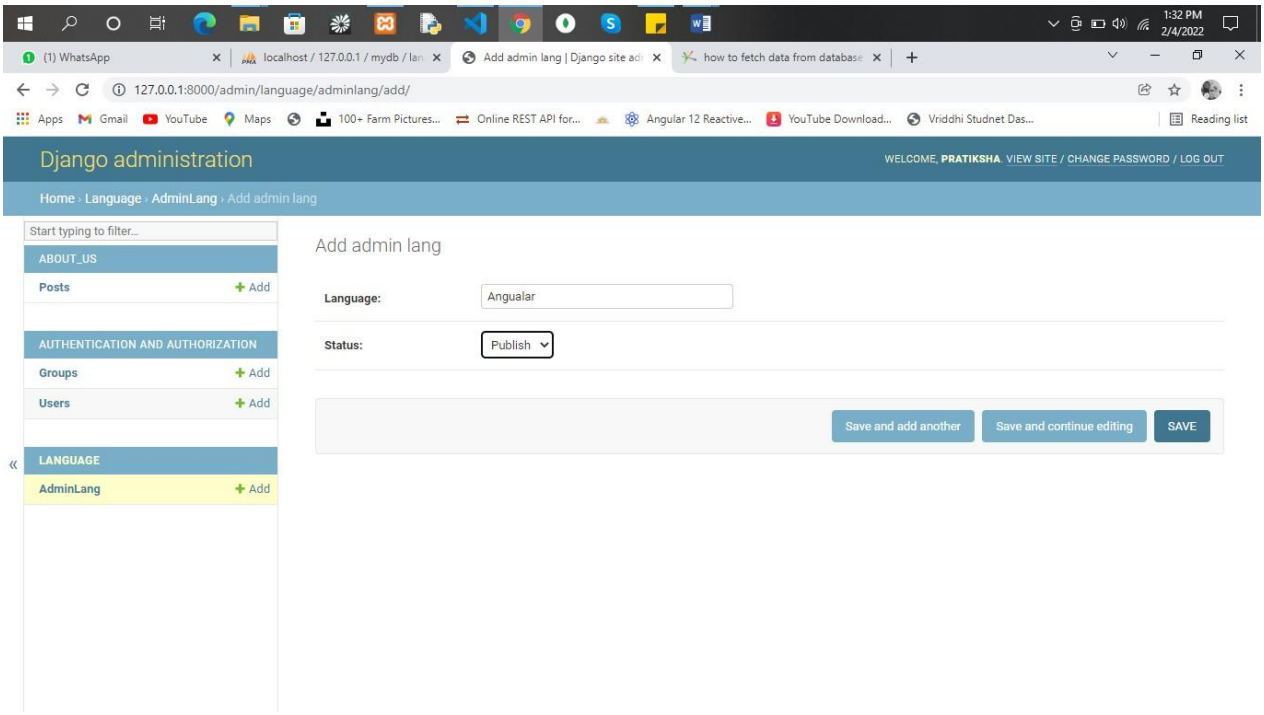
Models.py

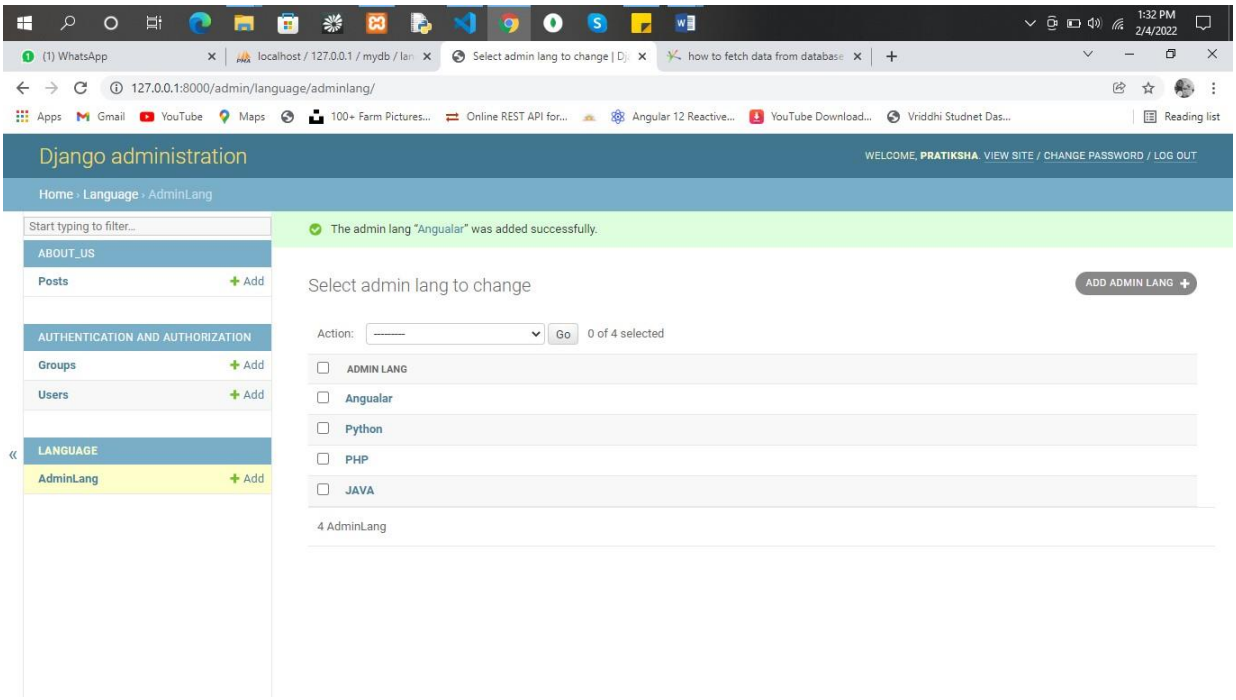
```
from django.db import models
from django.contrib.auth.models import User


# Create your models here. class
Language(models.Model):
    langname=models.CharField(max_length=63)

STATUS = (
    (0,"Draft"),
    (1,"Publish")
)
class AdminLang(models.Model):
    language=models.CharField(max_length=63)
    status = models.IntegerField(choices=STATUS, default=0)class
    Meta:
        verbose_name_plural = "AdminLang"

    def__str__(self): return
        self.language
```

admin.py

```
from django.contrib import admin

from language.models import AdminLang

@admin.register(AdminLang)
class AdminLangAdmin(admin.ModelAdmin):
    pass
```

Output :


Django administration
WELCOME, PRATIKSHA. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Language › AdminLang › Add admin lang
Start typing to filter...
ABOUT_US
Posts + Add
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
LANGUAGE
AdminLang + Add
Add admin lang
Language: Angular
Status: Publish
Save and add another
Save and continue editing
SAVE


Django administration
WELCOME, PRATIKSHA. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Language › AdminLang
Start typing to filter...
ABOUT_US
Posts + Add
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
LANGUAGE
AdminLang + Add
The admin lang "Angular" was added successfully.
Select admin lang to change
ADD ADMIN LANG +
Action: Go 0 of 4 selected
ADMIN LANG
Angular
Python
PHP
JAVA
4 AdminLang

Q. 1 Write a Java Program to implement State Pattern for Gumball Machine. Create instance variable that holds current state from there, we just need to handle all actions, behaviors andstate transition that can happen

1) Create Interface-

State.java

```
package javaprograms; public

interface State {

        public void insertQuarter();public
        void ejectQuarter(); public void
        turnCrank(); public void
        dispense();

        public void refill();
}
```

2) Create Class SoldState.java

```
package javaprograms;

public class SoldState implements State {

      GumballMachine gumballMachine;

      public SoldState(GumballMachine gumballMachine) {
            this.gumballMachine = gumballMachine;
      }

         public void insertQuarter() {
                  System.out.println("Please wait, we're already giving you a
gumball");
         }

         public void ejectQuarter() {
                  System.out.println("Sorry, you already turned the crank");
         }

         public void turnCrank() {
                        System.out.println("Turning twice doesn't get you another
gumball!");
         }

         public void dispense() {
                  gumballMachine.releaseBall();
                  if (gumballMachine.getCount() > 0) {
```

```
gumballMachine.setState(gumballMachine.getNoQuarterState());
```

```
                } else {
                        System.out.println("Oops, out of gumballs!");
                        gumballMachine.setState(gumballMachine.getSoldOutState());
                }
        }

        public void refill() { }

        public String toString() {
                return "dispensing a gumball";
        }
}
```

SoldOutState.java

```
package javaprograms;


public class SoldOutState implements State {
      GumballMachine gumballMachine;

      public SoldOutState(GumballMachine gumballMachine) {
          this.gumballMachine = gumballMachine;
      }

        public void insertQuarter() {
                System.out.println("You can't insert a quarter, the machine is
sold out");
        }

        public void ejectQuarter() {
                System.out.println("You can't eject, you haven't inserted aquarter yet");
        }

        public void turnCrank() {
                System.out.println("You turned, but there are no gumballs");
        }

        public void dispense() {
                System.out.println("No gumball dispensed");
        }

        public void refill() { gumballMachine.setState(gumballMachine.getNoQuarterState());
        }

        public String toString() {return
                "sold out";
        }
}
```

NoQuarterState.java

```
package javaprograms;

public class NoQuarterState implements State {
	GumballMachine gumballMachine;

	public NoQuarterState(GumballMachine gumballMachine) {
		this.gumballMachine = gumballMachine;
	}

	  public void insertQuarter() { System.out.println("You inserted a
			quarter");
			gumballMachine.setState(gumballMachine.getHasQuarterState());
	  }

	  public void ejectQuarter() {
				System.out.println("You haven't inserted a quarter");
	  }

	  public void turnCrank() {
				System.out.println("You turned, but there's no quarter");
	    }

	  public void dispense() {
			System.out.println("You need to pay first");
	  }

	  public void refill() { } public

	  String toString() {
			return "waiting for quarter";
	  }
}
```

HasQuarterState.java

```
package javaprograms;


public class HasQuarterState implements State {
	  GumballMachine gumballMachine;

	  public HasQuarterState(GumballMachine gumballMachine) {
			this.gumballMachine = gumballMachine;
	  }

	  public void insertQuarter() {
				System.out.println("You can't insert another quarter");
	  }

	  public void ejectQuarter() {
```

```
                System.out.println("Quarter returned");
                gumballMachine.setState(gumballMachine.getNoQuarterState());
        }

        public void turnCrank() { System.out.println("You
                turned...");
                gumballMachine.setState(gumballMachine.getSoldState());
        }

    public void dispense() {
        System.out.println("No gumball dispensed");
    }

    public void refill() { } public

        String toString() {
                return "waiting for turn of crank";
        }
}
```

GumballMachine.java

```
package javaprograms;


public class GumballMachine {

        State soldOutState; State
        noQuarterState; State
        hasQuarterState;State
        soldState;

        State state; int
        count = 0;

        public GumballMachine(int numberGumballs) { soldOutState
                = new SoldOutState(this); noQuarterState = new
                NoQuarterState(this); hasQuarterState = new
                HasQuarterState(this);soldState = new SoldState(this);

                this.count = numberGumballs;if
                (numberGumballs > 0) {
                        state = noQuarterState;
                } else {
                        state = soldOutState;
                }
        }

        public void insertQuarter() {
                state.insertQuarter();
```

```
        }

        public void ejectQuarter() {
                state.ejectQuarter();
        }

        public void turnCrank() {
                state.turnCrank();
                state.dispense();
        }

        void releaseBall() {
                System.out.println("A gumball comes rolling out the slot...");if (count > 0) {
                        count = count - 1;
                }
        }

        int getCount() {
                return count;
        }

        void refill(int count) { this.count
                += count;
                System.out.println("The gumball machine was just refilled; itsnew count is: "
+ this.count);
                state.refill();
        }

        void setState(State state) {this.state
                = state;
        }
    public State getState() {return
        state;
    }

    public State getSoldOutState() {return
        soldOutState;
    }

    public State getNoQuarterState() {return
        noQuarterState;
    }

    public State getHasQuarterState() {return
        hasQuarterState;
    }

    public State getSoldState() {return
        soldState;
```

```
    }

        public String toString() {
                StringBuffer result = new StringBuffer(); result.append("\nMighty Gumball,
                Inc."); result.append("\nJava-enabled Standing Gumball Model #2004");
                result.append("\nInventory: " + count + " gumball");
                if (count != 1) {
                        result.append("s");
                }
                result.append("\n");
                result.append("Machine is " + state + "\n");return
                result.toString();
        }
}
```

GumballMachineTestDrive.javapackage
javaprograms;

```
public class GumballMachineTestDrive {

        public static void main(String[] args) {
                GumballMachine gumballMachine = new GumballMachine(2);

                System.out.println(gumballMachine);

                gumballMachine.insertQuarter();
                gumballMachine.turnCrank();

                System.out.println(gumballMachine);

                gumballMachine.insertQuarter();
                gumballMachine.turnCrank();
                gumballMachine.insertQuarter();
                gumballMachine.turnCrank();

                gumballMachine.refill(5);
                gumballMachine.insertQuarter();
                gumballMachine.turnCrank();

                System.out.println(gumballMachine);
        }
}
```

Output-
Mighty Gumball, Inc.
Java-enabled Standing Gumball Model #2004
Inventory: 2 gumballs
Machine is waiting for quarter

You inserted a quarterYou
turned...
A gumball comes rolling out the slot...

Mighty Gumball, Inc.
Java-enabled Standing Gumball Model #2004
Inventory: 1 gumball
Machine is waiting for quarter

You inserted a quarterYou
turned...
A gumball comes rolling out the slot...Oops, out of
gumballs!
You can't insert a quarter, the machine is sold outYou turned, but
there are no gumballs
No gumball dispensed
The gumball machine was just refilled; its new count is: 5You inserted a
quarter
You turned...
A gumball comes rolling out the slot...

Mighty Gumball, Inc.
Java-enabled Standing Gumball Model #2004
Inventory: 4 gumballs
Machine is waiting for quarter

Q.2. Write a python program to find all null values in a given dataset and remove them.

```
# importing pandas as pdimport
pandas as pd

# importing numpy as npimport
numpy as np

# dictionary of lists
dict = {'First Score':[100, 90, np.nan, 95], 'Second Score': [30,
            45, 56, np.nan],
            'Third Score':[np.nan, 40, 80, 98]}

# creating a dataframe from listdf =
pd.DataFrame(dict)

# using isnull() function print("\n isnull()
function ");print(df.isnull())

# filling missing value using fillna() print("\n After filling null
values with 0");
```

```
print(df.fillna(0))
```

Q.3 Create your own blog using Django.

Base.html

```
<!DOCTYPE html>
<html>

    <head>
        <title>Django Central</title>
        <link href="https://fonts.googleapis.com/css?family=Roboto:400,700"rel="stylesheet">
        <meta name="google" content="notranslate" />
        <meta name="viewport" content="width=device-width, initial-scale=1" />
        <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
            crossorigin="anonymous" />
    </head>

    <body>
        <style>
            body {
            font-family: "Roboto", sans-serif;font-size:
            17px;
            background-color: #fdfdfd;
        }
        .shadow {
            box-shadow: 0 4px 2px -2px rgba(0, 0, 0, 0.1);
        }
        .btn-danger {
            color: #fff;
            background-color: #f00000;border-
            color: #dc281e;
        }
        .masthead {
            background: #3398E1;
            height: auto; padding-
            bottom: 15px;
            box-shadow: 0 16px 48px #E3E7EB;
            padding-top: 10px;
        }
```

```
    </style>

        <!-- Navigation -->
        <nav class="navbar navbar-expand-lg navbar-light bg-light shadow"id="mainNav">
            <div class="container-fluid">
                <a class="navbar-brand" href="{% url 'home' %}">Django
central</a>
                <button class="navbar-toggler navbar-toggler-right" type="button"
data-toggle="collapse" data-target="#navbarResponsive"
                    aria-controls="navbarResponsive" aria-expanded="false" aria-label="Toggle
navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="collapse navbar-collapse" id="navbarResponsive">
                    <ul class="navbar-nav ml-auto">
                        <li class="nav-item text-black">
                            <a class="nav-link text-black font-weight-bold"
href="#">About</a>
                        </li>
                        <li class="nav-item text-black">
                            <a class="nav-link text-black font-weight-bold"
href="#">Policy</a>
                        </li>
                        <li class="nav-item text-black">
                            <a class="nav-link text-black font-weight-bold"
href="#">Contact</a>
                        </li>
                    </ul>
                </div>
            </div>
        </nav>



        {% block content %}
        <!-- Content Goes here -->
        {% endblock content %}
        <!-- Footer -->
        <footer class="py-3 bg-grey">
            <p class="m-0 text-dark text-center ">Copyright &copy; Django
Central</p>
        </footer>
    </body>
</html>
```

Index.html

```
{% extends "base.html" %}
{% block content %}
<style>
	body {
		font-family: "Roboto", sans-serif;font-size:
		18px;
		background-color: #fdfdfd;
	}

	.head_text {
		color: white;
	}

	.card {
		box-shadow: 0 16px 48px #E3E7EB;
	}
</style>

<header class="masthead">
	<div class="overlay"></div>
	<div class="container">
		<div class="row">
			<div class=" col-md-8 col-md-10 mx-auto">
				<div class="site-heading">
					<h3 class=" site-heading my-4 mt-3 text-white"> Welcome to my
Awesome Blog </h3>
					{% comment %} <p class="text-light">We Love Django As much as
you do..!   {% endcomment %}
					</p>
				</div>
			</div>
		</div>
	</div>
</header>
<div class="container">
	<div class="row">
		<!-- Blog Entries Column -->
		<div class="col-md-8 mt-3 left">
			{% for post in post_list %}
			<div class="card mb-4">
				<div class="card-body">
					<h2 class="card-title">{{ post.title }}</h2>
					<p class="card-text text-muted h6">{{ post.author }} | {{post.created_on}}
</p>
```

```
                        <p class="card-text">{{post.content|slice:":200" }}</p>
                        <a href="{% url 'post_detail' post.slug %}" class="btn btn-primary">Read More
&rarr;</a>
                    </div>
                </div>
                {% endfor %}
            </div>
            {% block sidebar %} {% include 'sidebar.html' %} {% endblock sidebar %}
        </div>
</div>
{%endblock%}
```

Sidebar.html

```
{% block sidebar %}

<style>
        .card{
            box-shadow: 0 16px 48px #E3E7EB;
        }

</style>

<!-- Sidebar Widgets Column -->
<div class="col-md-4 float-right ">
<div class="card my-4">

</div>
</div>

{% endblock sidebar %}
```

Post_detail.html

```
{% extends 'base.html' %} {% block content %}

<div class="container">
    <div class="row">
        <div class="col-md-8 card mb-4   mt-3 left   top">
            <div class="card-body">
                <h1>{% block title %} {{ object.title }} {% endblock title %}</h1>
                <p class=" text-muted">{{ post.author }} | {{ post.created_on }}</p>
                <p class="card-text ">{{ object.content | safe }}</p>
            </div>
        </div>
        {% block sidebar %} {% include 'sidebar.html' %} {% endblock sidebar %}
    </div>
</div>{% endblock content %}
```

Admin.py

```
from django.contrib import adminfrom
.models import Post

class PostAdmin(admin.ModelAdmin):
        list_display = ('title', 'slug', 'status','created_on')list_filter = ("status",)
        search_fields = ['title', 'content'] prepopulated_fields = {'slug': ('title',)}

admin.site.register(Post, PostAdmin)
```

models.py

```
from django.db import models
from django.contrib.auth.models import User


STATUS = (
        (0,"Draft"),
        (1,"Publish")
)

class Post(models.Model):
        title = models.CharField(max_length=200, unique=True)slug =
        models.SlugField(max_length=200, unique=True) author =
        models.ForeignKey(User, on_delete=
models.CASCADE,related_name='blog_posts')
        updated_on = models.DateTimeField(auto_now= True)content =
        models.TextField()
        created_on = models.DateTimeField(auto_now_add=True) status =
        models.IntegerField(choices=STATUS, default=0)

        class Meta:
                ordering = ['-created_on']

        def str__(self): return
                self.title
```

views.py

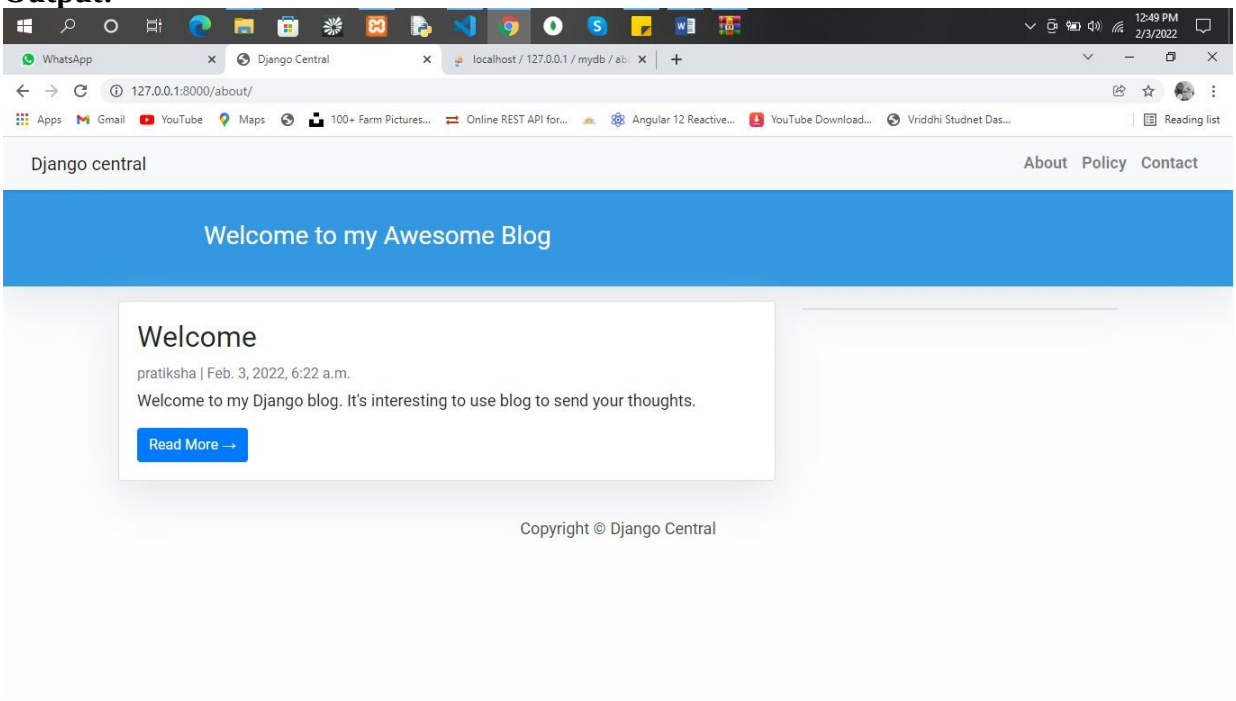
```
from django.views import genericfrom
.models import Post

class PostList(generic.ListView):
```

```
    queryset = Post.objects.filter(status=1).order_by('-created_on')template_name =
    'index.html'

class PostDetail(generic.DetailView):model = Post
    template_name = 'post_detail.html'
```

urls.py

```
from . import views
from django.urls import path

urlpatterns = [
    path('', views.PostList.as_view(), name='about'), path('<slug:slug>/',
    views.PostDetail.as_view(), name='post_detail'),
]
```

Output:



Q.1 Write a Java Program to implement Iterator Pattern for Designing Menu like Breakfast, Lunch or Dinner Menu

```
public interface Iterator {boolean
        hasNext(); Object next();
}


import java.util.Iterator;

public interface Menu {

        public Iterator<?> createIterator(); public static
        final String name = "";public String getName();

}
```

```
import java.util.Iterator;
public class DinerMenu implements Menu {static
        final int MAX_ITEMS = 6; int
        numberOfItems = 0;
        MenuItem[]
        menuItems;String
        name;

        public DinerMenu() {
                name="LUNCH";
                menuItems = new MenuItem[MAX_ITEMS];

                addItem("Vegetarian BLT",
                        "(Fakin') Bacon with lettuce & tomato on whole wheat", true,
2.99);
                addItem("BLT",
                        "Bacon with lettuce & tomato on whole wheat", false, 2.99);
                addItem("Soup of the day",
                        "Soup of the day, with a side of potato salad", false, 3.29);
                addItem("Hotdog",
                        "A hot dog, with saurkraut, relish, onions, topped with cheese",
                        false, 3.05);
                addItem("Steamed Veggies and Brown Rice",
                        "Steamed vegetables over brown rice", true, 3.99);
                addItem("Pasta",
                        "Spaghetti with Marinara Sauce, and a slice of sourdough bread",
                        true, 3.89);

        }

                public void addItem(String name, String description,
                                        boolean vegetarian, double price)
        {
                MenuItem menuItem = new MenuItem(name, description, vegetarian, price);
                if (numberOfItems >= MAX_ITEMS) {
```

```
                        System.err.println("Sorry, menu is full!  Can't add item to
menu");
                } else {
                        menuItems[numberOfItems] =
                        menuItem;numberOfItems =
                        numberOfItems + 1;
                }
        }

        public MenuItem[] getMenuItems() {
                return menuItems;
        }

        public Iterator<MenuItem> createIterator() {
                return new   DinerMenuIterator(menuItems);
        }

        @Override
        public String getName() {
                // TODO Auto-generated method stub
                return name;
        }

        // other menu methods here
}


import java.util.Iterator;
public class DinerMenuIterator implements Iterator {MenuItem[] list;
        int position = 0;

        public DinerMenuIterator(MenuItem[] list) {
                this.list = list;
        }
  public MenuItem next()
  {
           MenuItem menuItem=list[position];
           position=position+1;
           return menuItem;
  }


             public boolean hasNext() {
                        if (position >= list.length || list[position] == null) {

                           return false;
                     } else {

                           return true;
                }
        }
        public void remove()
        {
                if(position <=0) {
                        throw new IllegalStateException("You can't remove an item untilyou have
done least one next()");
```

```
                }
                if(list[position-1]!=null) {
                        for(int i=position-1;i < (list.length-1);i++) {
                                list[i]=list[i+1];
                        }
                        list[list.length-1]=null;
                }
        }
}



import java.util.ArrayList;

public class Main {

        public static void main(String[] args) {

                //DinerMenuIterator d=new DinerMenuIterator(MenuItem[] );
                PancakeHouseMenu pancakeHouseMenu=new
                PancakeHouseMenu(); DinerMenu dinerMenu = new
                DinerMenu();
                // MenuItem[] lunchItems = dinerMenu.getMenuItems();
                ArrayList<Menu> menus=new ArrayList<Menu>();
                menus.add(pancakeHouseMenu);
                menus.add(dinerMenu);
                Waitress waitress=new Waitress(menus);
                waitress.printMenu();
        }

}



public class MenuItem { String
        name; String description;
        boolean vegetarian;
        double price;

        public MenuItem(String name,
                        String description,
                        boolean vegetarian,
                        double price)
        {
                this.name = name; this.description
                = description;this.vegetarian =
                vegetarian; this.price = price;
        }

            public String getName() {
                  return name;
        }
```

```
        public String getDescription() {
                return description;
        }

        public double getPrice() {
                return price;
        }

        public boolean isVegetarian() {
                return vegetarian;
        }
        public String toString() {
                return (name + ", $" + price + "\n            " + description);
        }
}


import java.util.ArrayList;
import java.util.Iterator;
public class PancakeHouseMenu implements Menu {
        ArrayList menuItems;
        String name;

        public PancakeHouseMenu() {
                name="BREAKFAST";
                menuItems = new ArrayList();

                addItem("K & B's Pancake Breakfast",
                        "Pancakes with scrambled eggs, and toast",
                        true,
                        2.99);

                addItem("Regular Pancake Breakfast",
                        "Pancakes with fried eggs, sausage",
                        false,
                        2.99);

                addItem("Blueberry Pancakes",
                        "Pancakes made with fresh blueberries",
                        true,
                        3.49);

                addItem("Waffles",
                        "Waffles, with your choice of blueberries or strawberries",
                        true,
                        3.59);
        }

        public void addItem(String name, String description,
```

```
                    boolean vegetarian, double price)
        {
            MenuItem menuItem = new MenuItem(name, description, vegetarian, price);
            menuItems.add(menuItem);
        }

        public ArrayList<MenuItem> getMenuItems() {
            return menuItems;
        }

        public Iterator<MenuItem> createIterator() {
            return menuItems.iterator();
        }

        public String toString() {
            return "Objectville Pancake House Menu";
        }
        public String getName() {
            return name;}

        // other menu methods here
}
```

```
import java.util.ArrayList;

public class PancakeHouseMenuIterator implements Iterator {ArrayList items;
int position = 0;

public PancakeHouseMenuIterator(ArrayList items) {
this.items = items;
}

public Object next() {
Object object = items.get(position);
position = position + 1;
return object;
}

public boolean hasNext() {
if (position >= items.size()) {

return false;
} else {

return true;
}
}

public ArrayList getMenuItems() {
        // TODO Auto-generated method stub
        return null;
```

```
}}
```

```
import java.util.ArrayList;
import java.util.Iterator;
public class Waitress {
	ArrayList<Menu> menus;

	public Waitress(ArrayList<Menu> menus) {
		this.menus=menus;
	}

	public void printMenu() {
		Iterator <?> menuIterator=menus.iterator();
		System.out.println("MENU\n ---\n\n");

		while(menuIterator.hasNext())
		{
			Menu menu=(Menu)menuIterator.next();
			System.out.println("\n"+menu.getName()+"\n");
			printMenu(menu.createIterator());

		}

	}

	 void printMenu(Iterator<?> iterator) {
		while (iterator.hasNext()) {
			MenuItem menuItem = (MenuItem)iterator.next();
			System.out.print(menuItem.getName() + ", ");
			System.out.print(menuItem.getPrice() + " -- ");
			System.out.println(menuItem.getDescription());
		}
	}

}
```

Output :

```
MENU
----

BREAKFAST

K & B's Pancake Breakfast, 2.99 -- Pancakes with scrambled eggs, and toastRegular
Pancake Breakfast, 2.99 -- Pancakes with fried eggs, sausage Blueberry Pancakes,
3.49 -- Pancakes made with fresh blueberries
Waffles, 3.59 -- Waffles, with your choice of blueberries or strawberries
```

LUNCH

Vegetarian BLT, 2.99 -- (Fakin') Bacon with lettuce & tomato on whole wheatBLT, 2.99 -- Bacon with lettuce & tomato on whole wheat
Soup of the day, 3.29 -- Soup of the day, with a side of potato salad Hotdog, 3.05 -- A hot dog, with saurkraut, relish, onions, topped with cheeseSteamed Veggies and Brown Rice, 3.99 -- Steamed vegetables over brown rice Pasta, 3.89 -- Spaghetti with Marinara Sauce, and a slice of sourdough bread

Q.2. Write a python program to make Categorical values in numeric format for a givendataset.

```
#import pandas import
pandas as pd

# read csv file
df = pd.read_csv('Customers.csv')print(df)
print("\n After repalcing Category Male as 0 and Female as 1");# replacing values

df['GENDER'].replace(['Male', 'Female'],
                                        [0, 1], inplace=True)
print(df)
```

Output:

```
  Name  Episodes  Gender
0  Sheldon        42    male
1     Penny       24 female
2       Amy       31 female
3     Penny       29 female
4       Raj       37    male
5  Sheldon        40    male
      Name  Episodes  Gender  female  male
0  Sheldon        42    male       0     1
1     Penny       24 female        1     0
2       Amy       31 female        1     0
3     Penny       29 female        1     0
4       Raj       37    male       0     1
5  Sheldon        40    male       0     1
```

Q.3 Implement Login System using Django.

home app

forms.py

```
from django import forms class

LoginForm(forms.Form):
```

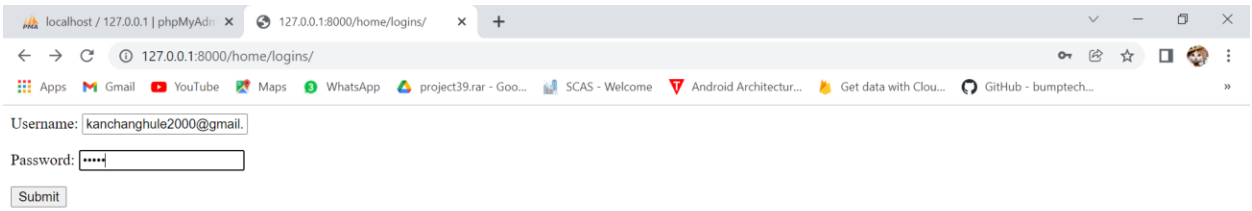
```
    username=forms.CharField(max_length=63)
    password=forms.CharField(max_length=63,widget=forms.PasswordInput)
```

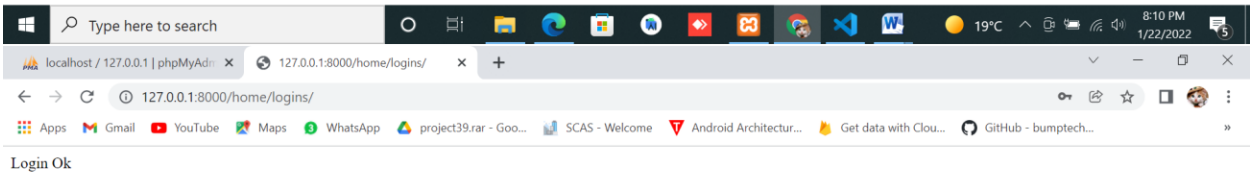
login.html

```
<form method="post">
    {{ form.as_p }}
    {% csrf_token %}
    <button type="submit">Submit</button>
</form>
```

views.py

```
from django.shortcuts import renderfrom .
import forms

def login(request): form=forms.LoginForm()
    if request.method=='POST':
        form=forms.LoginForm(request.POST)if
        form.is_valid():
            return HttpResponse("Login Ok")
    return render(request,'login.html',context={'form':form})urls.py

from django.urls import pathfrom
.views import login urlpatterns = [
    path('logins/',login, name="home"),
]
test_project
from django.contrib import adminfrom
django.urls import path from django.urls
import include urlpatterns = [
    path('home/', include("home.urls")),
]
```





Q.1 Write a Java Program to implement Singleton pattern for multithreading

```
package javaprograms;
public class SingletoneTest

{
        private static final int
                                                PROCESSOR_COUN
T =Runtime.getRuntime().availableProcessors();
        private static final Thread[]                   THREADS             = new
```

```
Thread[PROCESSOR_COUNT];
    private static int                  instancesCount      = 0;
    private static SingletoneTest       instance            = null;

     /*** private constructor to prevent Creation of Object from
 Outside of the * This class.
      */
    private SingletoneTest()
{
    }
/*** return the instance only if it does not exist */
    public static SingletoneTest getInstance()
{
        if (instance == null)
        {
            instancesCount++;
            instance = new SingletoneTest();
        }
        return instance;
}
    /*** reset instancesCount and instance.*/
    private static void reset()
{
        instancesCount = 0;
        instance = null;
}
    /*** validate system to run the test*/
    private static void validate()
{
        if (SingletoneTest.PROCESSOR_COUNT < 2)
        {
 test.");            System.out.print("PROCESSOR_COUNT Must be >= 2 to Run the

            System.exit(0);
        }
    }
    public static void main(String... args)
{
        validate();
```

```
        System.out.printf("Summary :: PROCESSOR_COUNT %s, Running Testwith %s of
Threads. %n", PROCESSOR_COUNT, PROCESSOR_COUNT);
        long currentMili = System.currentTimeMillis();
        int testCount = 0;
        do

        {
            reset();
            for (int i = 0; i < PROCESSOR_COUNT; i++)
                THREADS[i] = new Thread(SingletoneTest::getInstance);

            for (int i = 0; i < PROCESSOR_COUNT; i++)
                THREADS[i].start();

            for (int i = 0; i < PROCESSOR_COUNT; i++)
                try

                {
                    THREADS[i].join();
                }
                catch (InterruptedException e)

                {
                    e.printStackTrace();
                    Thread.currentThread().interrupt();

                }
            testCount++;

        }
        while (instancesCount <= 1 && testCount < Integer.MAX_VALUE);


        System.out.printf("Singleton Pattern is broken after %d try.
%nNumber of instances count is %d. %nTest duration %dms", testCount,
instancesCount, System.currentTimeMillis() - currentMili);

}
 }
```

Output-

Summary :: PROCESSOR_COUNT 4, Running Test with 4 of Threads.Singleton
Pattern is broken after 144 try.
Number of instances count is 2.Test
duration 232ms

Q.2. Write a python program to Implement Simple Linear Regression for predicting house price.

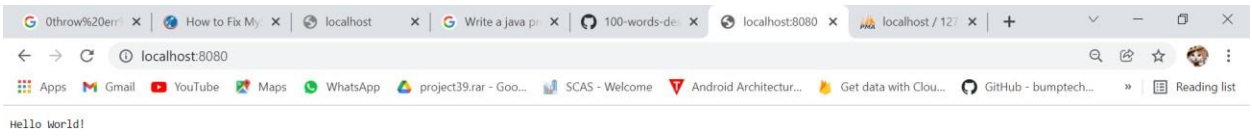
```
import pandas as pd
import numpy as np
```

```
import matplotlib.pyplot as pltimport
seaborn as sns
```

```
#sns.set_style("whitegrid")
#plt.style.use("fivethirtyeight")

USAhousing = pd.read_csv('USA_Housing.csv')
USAhousing.head()

X = USAhousing[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number ofRooms',
                'Avg. Area Number of Bedrooms', 'Area Population']]y =
USAhousing['Price']

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,random_state=101)



from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression(normalize=True)
lin_reg.fit(X_train,y_train)


pred = lin_reg.predict(X_test)
plt.scatter(y_test, pred) plt.show()
```

Q.3 Create a Simple Web Server using node js.

```
var http = require('http');
//create a server object:
http.createServer(function (req, res) {
   res.write('Hello World!'); //write a response to the clientres.end(); //end
   the response
}).listen(8080); //the server object listens on port 8080
```



Q.1 Write a Java Program to implement Strategy Pattern for Duck Behavior. Create instance variable that holds current state of Duck from there, we just need to handle all Flying Behaviors and Quack Behavior.

1) Create Interface

QuackBehaviour.java

```
package javaprograms;
public interface QuackBehaviour {
        public default void quack() {
            System.out.println("Quack");
        }
}
```

FlyBehaviour.java

```
package javaprograms;

public interface FlyBehaviour {
        public void fly();
}
```

2) Create Class -

FlyWithWings.java

```
package javaprograms;
public class FlyWithWings implements FlyBehaviour {
        public void fly() { System.out.println("I'm flying!!");
        }
}
```

Quack.java

```
package javaprograms;

public class Quack implements QuackBehaviour {
        public void quack() { System.out.println("Quack");
        }
}
```

ModolDuck.java

```
package javaprograms;
public class ModolDuck extends Duck {
        public ModolDuck() { flyBehaviour = new
            FlyNoWay(); quackBehaviour = new
            Quack();
        }

        public void display() { System.out.println("I'm a model
            duck");
        }
}
```

MallardDuck.java

```
package javaprograms;
```

```
public class MallardDuck extends Duck {

        public MallardDuck() { quackBehaviour = new
            Quack(); flyBehaviour = new
            FlyWithWings();
        }

        public void display() {
            System.out.println("I'm a real Mallard duck");
        }
}
```

Duck.java

```
package javaprograms;

public class MallardDuck extends Duck {

        public MallardDuck() { quackBehaviour = new
            Quack(); flyBehaviour = new
            FlyWithWings();
        }

        public void display() {
            System.out.println("I'm a real Mallard duck");
        }
}
```

FlyRocketPowered.java

```
package javaprograms;
public class FlyRocketPowered implements FlyBehaviour {
        public void fly() {
            System.out.println("I'm flying with a rocket!");
        }
}
```

FlyNoWay.java

```
package javaprograms;
public class FlyNoWay implements FlyBehaviour {
        public void fly() { System.out.println("I can't fly");
        }
}
```

MiniDuckSimulator.java

```
package javaprograms;

public class MiniDuckSimulator {
        public static void main(String[] args) { Duck mallard
            = new MallardDuck(); mallard.performQuack();
            mallard.performFly();
            Duck model = new ModolDuck();

            model.performFly();
```

```
                model.setFlyBehaviour(new FlyRocketPowered());

                model.performFly();
            }
}
```

Output-
Quack
I'm flying!!
I can't fly
I'm flying with a rocket!

Q.2. Write a python program to implement Multiple Linear Regression for given dataset.

```
import matplotlib.pyplot as plt import
pandas as pd
import numpy as np import
pylab as pl


df = pd.read_csv('Fuelconsumption.csv') df.head()

cdf =
df[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_CITY','FUELCONSUMPTION_HWY','FUELCO NSUMPTION_COMB','CO2EMISSIONS']]
cdf.head()

plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS, color='blue')
plt.xlabel('Engine Size')
plt.ylabel('Emissions') plt.show()

msk = np.random.rand(len(df)) < 0.8 train =
cdf[msk]
test = cdf[~msk]


from sklearn import linear_model
regr = linear_model.LinearRegression()
x = np.asanyarray(train[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
y = np.asanyarray(train[['CO2EMISSIONS']])
regr.fit(x,y)

print('Coefficients: ', regr.coef_)
```

Q.3 Create a Node.js file that demonstrates create database and table in MySQL.

```
var mysql = require('mysql');

var con = mysql.createConnection({ host:
   "localhost",
   user: "root", password: ""
});

con.connect(function(err) { if (err)
   throw err;
   console.log("Connected!");
   con.query("CREATE DATABASE mydb", function (err, result) { if
      (err) throw err;
      console.log("Database created");
   });
});
```

Q.1 Write a Java Program to implement Abstract Factory Pattern for Shape interface.

1) Interface Shape.java

```
package javaprograms;
public interface Shape {
        void draw();
}
```

Color.java

```
package javaprograms;
public interface Color {
        void fill();
}
```

2) Create Class

Rectangle.java package

```
javaprograms;

public class Rectangle implements Shape {@Override
        public void draw() {
        System.out.println("Inside Rectangle::draw() method.");
        }
}
```

Sqaure.java

```
package javaprograms;
```

```
public class Square implements Shape {
        @Override
        public void draw() {
        System.out.println("Inside Square::draw() method.");
        }
}
```

Circle.java

```
package javaprograms;
public class Circle implements Shape {@Override
        public void draw() {
        System.out.println("Inside Circle::draw() method.");
        }
}
```

Red.java

```
package javaprograms;
public class Red implements Color {
        @Override
        public void fill() {
        System.out.println("Inside Red::fill() method.");
        }
}
```

Green.java

```
package javaprograms;

public class Green implements Color {
        @Override
        public void fill() {
        System.out.println("Inside Green::fill() method.");
        }
}
```

Blue.java

```
package javaprograms;

public class Blue implements Color {
        @Override
        public void fill() {
        System.out.println("Inside Blue::fill() method.");
        }
}
```

AbstractFactory.java

```
package javaprograms;

public abstract class AbstractFactory { abstract Color
        getColor(String color); abstract Shape
        getShape(String shape) ;
       }
```

ShapeFactory.java

```
package javaprograms;
public class ShapeFactory extends AbstractFactory {@Override
        public Shape getShape(String shapeType){

        if(shapeType == null){return
        null;
        }
```

```
        if(shapeType.equalsIgnoreCase("CIRCLE")){return
        new Circle();

        }else if(shapeType.equalsIgnoreCase("RECTANGLE")){return
        new Rectangle();

        }else if(shapeType.equalsIgnoreCase("SQUARE")){return
        new Square();
        }

        return null;
        }

        @Override
        Color getColor(String color) {
        return null;
        }
}
```

ColorFactory.java

```
package javaprograms;

public class ColorFactory extends AbstractFactory {@Override
        public Shape getShape(String shapeType){
        return null;
        }

        @Override
        Color getColor(String color) {

        if(color == null){return
        null;
        }

        if(color.equalsIgnoreCase("RED")){return
        new Red();

        }else if(color.equalsIgnoreCase("GREEN")){return new
        Green();

        }else if(color.equalsIgnoreCase("BLUE")){return new
        Blue();
        }
```

```
        return null;
        }
      }
```

FactoryProducer.java

```
package javaprograms;

public class FactoryProducer {
        public static AbstractFactory getFactory(String choice){

        if(choice.equalsIgnoreCase("SHAPE")){return
        new ShapeFactory();

        }else if(choice.equalsIgnoreCase("COLOR")){return new
        ColorFactory();
        }

        return null;
        }
      }
```

AbstractFactoryPatternDemo.java

```
package javaprograms;

public class AbstractFactoryPatternDemo {
        public static void main(String[] args) {
        //get shape factory AbstractFactory
        shapeFactory =
FactoryProducer.getFactory("SHAPE");
        //get an object of Shape Circle
        Shape shape1 = shapeFactory.getShape("CIRCLE");
        //call draw method of Shape Circle
        shape1.draw();
        //get an object of Shape Rectangle
        Shape shape2 = shapeFactory.getShape("RECTANGLE");
        //call draw method of Shape Rectangle
        shape2.draw();

        //get an object of Shape Square
        Shape shape3 = shapeFactory.getShape("SQUARE");
        //call draw method of Shape Square
        shape3.draw();
        //get color factory AbstractFactory
        colorFactory =
FactoryProducer.getFactory("COLOR");
```

```
        //get an object of Color Red
        Color color1 = colorFactory.getColor("RED");
        //call fill method of Red
        color1.fill();
        //get an object of Color Green
        Color color2 = colorFactory.getColor("Green");
        //call fill method of Green
        color2.fill();
        //get an object of Color Blue
        Color color3 = colorFactory.getColor("BLUE");
        //call fill method of Color Bluecolor3.fill();
        }

}
```

Output

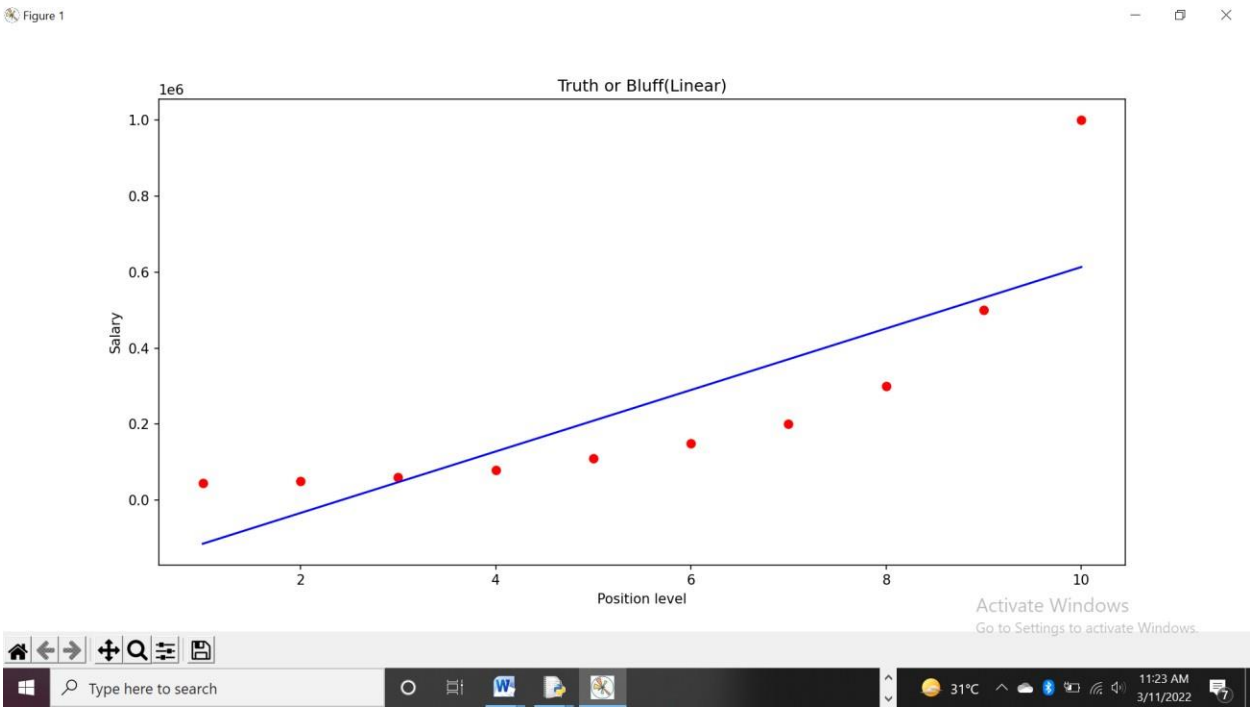
-

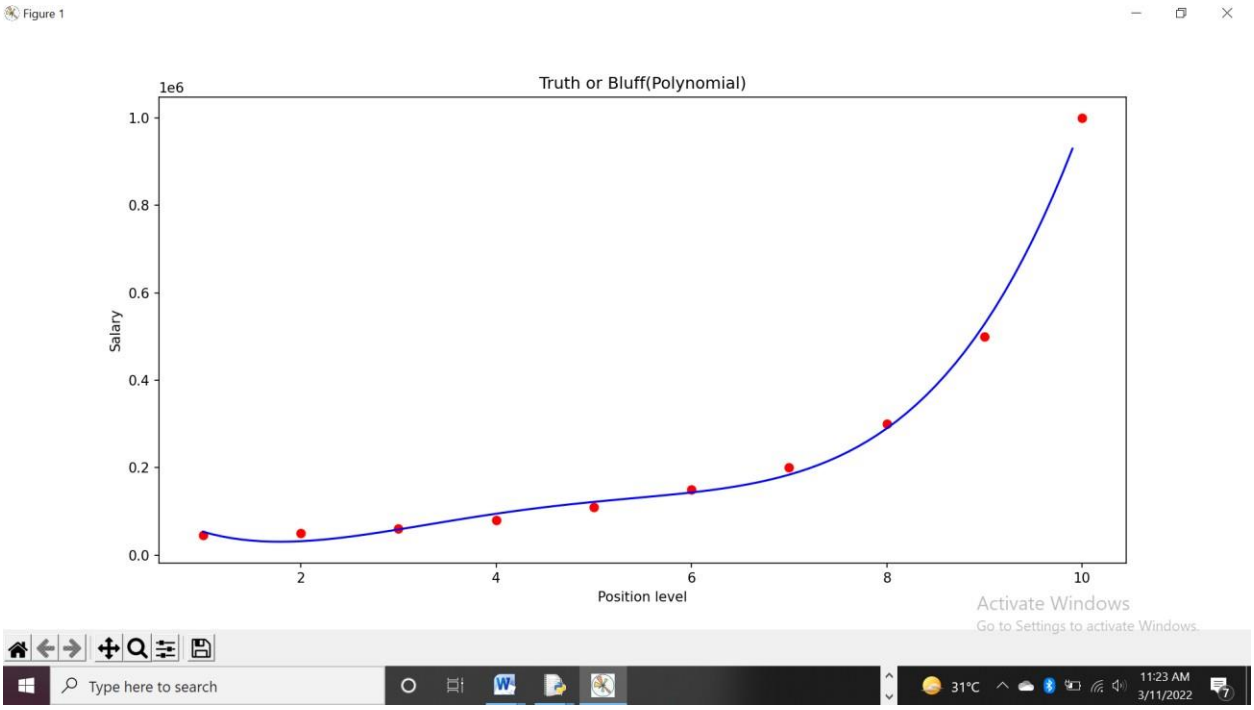
```
Inside Circle::draw() method. Inside
Rectangle::draw() method.Inside
Square::draw() method.
Inside Red::fill() method. Inside
Green::fill() method.Inside Blue::fill()
method.
```

Q.2. Write a python program to implement Polynomial Linear Regression for given dataset

```
import numpy as np
import matplotlib.pyplot as pltimport
pandas as pd

dataset = pd.read_csv('position_salaries.csv')dataset

X = dataset.iloc[:,1:2].valuesy =
dataset.iloc[:,2].values

# fitting the linear regression model
from sklearn.linear_model import LinearRegressionlin_reg =
LinearRegression()
lin_reg.fit(X,y)

# visualising the linear regression modelplt.scatter(X,y,
color='red')
plt.plot(X, lin_reg.predict(X),color='blue')plt.title("Truth or
Bluff(Linear)") plt.xlabel('Position level') plt.ylabel('Salary')
plt.show()
```

```
# polynomial regression model
from sklearn.preprocessing import PolynomialFeaturespoly_reg =
PolynomialFeatures(degree=2)
X_poly = poly_reg.fit_transform(X)

X_poly          # prints X_poly

lin_reg2 = LinearRegression()
lin_reg2.fit(X_poly,y)


# visualising polynomial regression
from sklearn.preprocessing import PolynomialFeaturespoly_reg =
PolynomialFeatures(degree=4)
X_poly = poly_reg.fit_transform(X)lin_reg2 =
LinearRegression() lin_reg2.fit(X_poly,y)

X_grid = np.arange(min(X),max(X),0.1) X_grid =
X_grid.reshape(len(X_grid),1)plt.scatter(X,y,
color='red')

plt.plot(X_grid, lin_reg2.predict(poly_reg.fit_transform(X_grid)),color='blue')plt.title("Truth or

Bluff(Polynomial)")
plt.xlabel('Position level')
plt.ylabel('Salary')plt.show()
```





Q.3 Create your Django app in which after running the server, you should see on the browser,the text “Hello! I am learning Django”, which you defined in the index view.
home app
views.py

```
from django.shortcuts import render
```

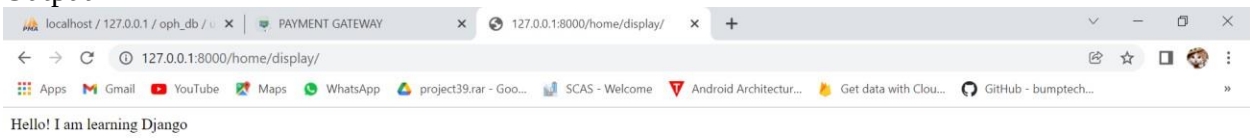
```
# Create your views here.
from django.http import HttpResponsedef
display(request):
        return HttpResponse("Hello! I am learning Django")urls.py
from django.urls import pathfrom
.views import display urlpatterns = [
        path('display/',display, name="home")
]
```

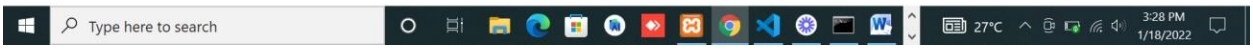
test_project

urls.py

```
from django.contrib import adminfrom
django.urls import path from django.urls
import include urlpatterns = [
        path('admin/', admin.site.urls), path('home/',
        include("home.urls")),
]
```

Output





Q.1 Write a JAVA Program to implement built-in support (java.util.Observable) Weather station with members temperature, humidity, pressure and methods mesurmentsChanged(),setMesurment(), getTemperature(), getHumidity(), getPressure()

1.Create

Interface

Observer.java

```
package javaprograms;

public interface Observer {
```

```
	public void update(float temp,float humidity,float pressure);
}
```

Displayelement.java

```
package hello;

public interface DisplayElement {

	public void display();
}
```

Subject.java

```
package hello;

public interface Subject {

	public void registerObserver(Observer o);
	public void removeObserver(Observer o);
	public void notifyObservers();
}
```

2. create classes

CurrentConditionDispaly.java

```
package hello;


public class CurrentConditionDispaly implements Observer,DisplayElement {


	private float temprature;

	private float humidity;

	private Subject  weatherData;


	public CurrentConditionDispaly(Subject weatherData)
{
		this.weatherData=weatherData;

		weatherData.registerObserver(this);



}


	public void update(float temprature,float humidity,float pressure) {
		this.temprature=temprature;

		this.humidity=humidity;

		display();

	}

	public void display()

	{
```

```
            System.out.println("current conditions:"+temprature+"F degree and
"+humidity+"% humidity");
      }



}
```

ForecastDisplay.java

```
package hello;

public class ForecastDisplay implements Observer,DisplayElement {

      private float  currentpressure=29.92f;
      private float lastpressure;
      private WeatherData weatherData;

      public ForecastDisplay(WeatherData weaherdata) {
            this.weatherData=weatherData;
            weatherData.registerObserver(this);
      }
      public void update(float temp,float humidity,float pressure) {
            lastpressure=currentpressure;
            currentpressure=pressure;
            display();
      }
```

```
	public void display()
	{
		System.out.println("forecast:");
		if(currentpressure > lastpressure) {
			System.out.println("improving weather on the way!.");
		}else if(currentpressure==lastpressure) {
			System.out.println("more of the same");
		}else if(currentpressure < lastpressure) {
			System.out.println("watch out for cooler,rainy weather");
		}

	}

}
```

HeatIndexDisplay.java

```
package hello;

public class HeatIndexDisplay implements Observer,DisplayElement {
	float heatIndex=0.0f;
	private WeatherData weatherData;
	public HeatIndexDisplay(WeatherData weatherData) {
		this.weatherData=weatherData;
		weatherData.registerObserver(this);
	  }
	public void update(float t,float rh,float pressure) {
		heatIndex=computeHeatIndex(t,rh);
```

```
            display();


      }
      private float computeHeatIndex(float t,float rh) {
            float index=(float)((16.923 + (0.185212 * t) + (5.37941 * rh) -
(0.100254 * t * rh) + (0.000345372 *(t * t * rh ))) +(0.00728898 * (rh * rh))
+ (0.000345372 * (t * t * rh)) - (0.000814971 * (t * rh * rh))+
(0.0000102102 * (t * t * rh * rh)) -(0.000038646 * (t * t * t)) +
(0.0000291683 * (rh * rh * rh)) + (0.00000142721 * (t * t * t * rh )) +
(0.000000197483 * (t * rh * rh * rh)) - (0.000000218429 * (t * t * t * rh *
rh )) + (0.000000000843296 * (t * t * rh * rh * rh)) - (0.0000000000481975 *
(t * t * t * rh * rh * rh)));
      return index;
      }
      public void display()
      {
            System.out.println("heat index"+heatIndex);
      }
}
```

StatisticDisplay.java

```
package hello;


public class StatisticDisplay  implements Observer,DisplayElement {
       private float maxTemp=0.0f;
       private float minTemp=200;
       private float tempSum=0.0f;
       private int numReadings;
     private WeatherData weatherData;
```

```
    public StatisticDisplay(WeatherData weatherData) {
      this.weatherData=weatherData;
      weatherData.registerObserver(this);
    }

    public void update(float temp,float humidity,float pressure)
    {
      tempSum=temp;
      numReadings++;

      if(temp > maxTemp)
      {
            maxTemp=temp;
     }
      if(temp < minTemp) {
            minTemp=temp;
      }
      display();
    }

    public void display()
    {
      System.out.println("AVG?MIN?MAX temprature="+(tempSum/numReadings
)+"/"+maxTemp+"/"+minTemp);

    }
}
```

WeatherData.java

```
package hello;

import java.util.ArrayList;

public class WeatherData implements Subject{
	private ArrayList<Observer> observers;
	private float temprature;
	private  float humidity;
	private float pressure;


	public WeatherData() {
		observers=new ArrayList<>();
	}

	public void registerObserver(Observer o) {
		observers.add(o);
	}

	public void removeObserver(Observer o) {
		int i=observers.indexOf(o);
		if(i>=0) {
			observers.remove(i);
		}
```

```
	}

public void notifyObservers() {
	for(int i=0;i<observers.size();i++) {
		Observer observer=(Observer)observers.get(i);
		observer.update(temprature, humidity, pressure);
	}

}
public void measurementChanged() {
	notifyObservers();
}
public void setMeasurement(float temprature,float humidity,float pressure) {
this.temprature=temprature;
this.humidity=humidity;
this.pressure=pressure;
measurementChanged();
}
public float getTemprature()
{
  return temprature;
}
public float gethumidity()
{
return  humidity;
}
```

```
public float getpressure()
{
return pressure;
}
}
```

WeatherStation.java

```
package hello;
import  java.io.*;
public class WeatherStation {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		//try {
		WeatherData weatherData=new WeatherData();
		CurrentConditionDispaly currentDisplay=new
CurrentConditionDispaly(weatherData);
		StatisticDisplay statisticDisplay=new
StatisticDisplay(weatherData);
		weatherData.setMeasurement(80,65,30.4f);
		weatherData.setMeasurement(82, 70,29.2f);
		weatherData.setMeasurement(78,90,29.2f);
	}
}
```

Output-

current conditions:80.0F degree and 65.0% humidity

AVG?MIN?MAX temprature=80.0/80.0/80.0

```
current conditions:82.0F degree and 70.0% humidity

AVG?MIN?MAX temprature=41.0/82.0/80.0

current conditions:78.0F degree and 90.0% humidity

AVG?MIN?MAX temprature=26.0/82.0/78.0
```

Q.2. Write a python program to implement Naive Bayes.

```
from sklearn import datasets
from sklearn import metrics
from sklearn.naive_bayes import GaussianNB



dataset = datasets.load_iris()

#Creating our Naive Bayes Model
model = GaussianNB()
model.fit(dataset.data, dataset.target)

#Making Predictions
expected = dataset.target
predicted = model.predict(dataset.data)

#Getting Accuracy and Statistics
print(metrics.classification_report(expected, predicted))
print(metrics.confusion_matrix(expected, predicted))
```

Q.3 Create your own blog using

DjangoBase.html

```
<!DOCTYPE html>
<html>

  <head>
    <title>Django Central</title>
    <link href="https://fonts.googleapis.com/css?family=Roboto:400,700" rel="stylesheet">
    <meta name="google" content="notranslate" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS
6JXm"
```

```
        crossorigin="anonymous" />
    </head>

    <body>
      <style>
        body {
        font-family: "Roboto", sans-serif;
        font-size: 17px;
        background-color: #fdfdfd;
      }
      .shadow {
        box-shadow: 0 4px 2px -2px rgba(0, 0, 0, 0.1);
      }
      .btn-danger {
        color: #fff;
        background-color: #f00000;
        border-color: #dc281e;
      }
      .masthead  {
        background: #3398E1;
        height: auto;
        padding-bottom: 15px;
        box-shadow: 0 16px 48px
        #E3E7EB;padding-top: 10px;
      }
    </style>

      <!-- Navigation -->
      <nav class="navbar navbar-expand-lg navbar-light bg-light shadow" id="mainNav">
        <div class="container-fluid">
          <a class="navbar-brand" href="{% url 'home' %}">Django central</a>
          <button class="navbar-toggler navbar-toggler-right" type="button" data-
toggle="collapse" data-target="#navbarResponsive"
            aria-controls="navbarResponsive" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarResponsive">
            <ul class="navbar-nav ml-auto">
              <li class="nav-item text-black">
                <a class="nav-link text-black font-weight-bold" href="#">About</a>
              </li>
              <li class="nav-item text-black">
                <a class="nav-link text-black font-weight-bold" href="#">Policy</a>
```

```
                </li>
                <li class="nav-item text-black">
                  <a class="nav-link text-black font-weight-bold" href="#">Contact</a>
                </li>
              </ul>
            </div>
          </div>
        </nav>


        {% block content %}
        <!-- Content Goes here -->
        {% endblock content %}
        <!-- Footer -->
        <footer class="py-3 bg-grey">
          <p class="m-0 text-dark text-center ">Copyright &copy; Django Central</p>
        </footer>
    </body>
</html>
```

Index.html

```
{% extends "base.html" %}
{% block content %}
<style>
    body {
        font-family: "Roboto", sans-
        serif;font-size: 18px;
        background-color: #fdfdfd;
    }

    .head_text {
        color: white;
    }

    .card {
        box-shadow: 0 16px 48px #E3E7EB;
    }
</style>
```

```
<header class="masthead">
   <div class="overlay"></div>
   <div class="container">
      <div class="row">
         <div class=" col-md-8 col-md-10 mx-auto">
            <div class="site-heading">
               <h3 class=" site-heading my-4 mt-3 text-white"> Welcome to my Awesome Blog
</h3>
               {% comment %} <p class="text-light">We Love Django As much as you do..!  
{% endcomment %}
               </p>
            </div>
         </div>
      </div>
   </div>
</header>
<div class="container">
   <div class="row">
      <!-- Blog Entries Column -->
      <div class="col-md-8 mt-3 left">
         {% for post in post_list %}
         <div class="card mb-4">
            <div class="card-body">
               <h2 class="card-title">{{ post.title }}</h2>
               <p class="card-text text-muted h6">{{ post.author }} | {{ post.created_on}} </p>
               <p class="card-text">{{post.content|slice:":200" }}</p>
               <a href="{% url 'post_detail' post.slug  %}" class="btn btn-primary">Read More
&rarr;</a>
            </div>
         </div>
         {% endfor %}
      </div>
      {% block sidebar %} {% include 'sidebar.html' %} {% endblock sidebar %}
   </div>
</div>
{%endblock%}
```

Sidebar.html

```
{% block sidebar %}

<style>
      .card{
         box-shadow: 0 16px 48px #E3E7EB;
```

```
    }

</style>

<!-- Sidebar Widgets Column -->
<div class="col-md-4 float-right ">
<div class="card my-4">

</div>
</div>

{% endblock sidebar %}
```

Post_detail.html

```
{% extends 'base.html' %} {% block content %}

<div class="container">
  <div class="row">
    <div class="col-md-8 card mb-4  mt-3 left  top">
      <div class="card-body">
        <h1>{% block title %} {{ object.title }} {% endblock title %}</h1>
        <p class=" text-muted">{{ post.author }} | {{ post.created_on }}</p>
        <p class="card-text ">{{ object.content | safe }}</p>
      </div>
    </div>
    {% block sidebar %} {% include 'sidebar.html' %} {% endblock sidebar %}
            </div>
</div>

{% endblock content %}
```

Admin.py

```
from django.contrib import admin
from .models import Post

class PostAdmin(admin.ModelAdmin):
    list_display = ('title', 'slug', 'status','created_on')list_filter
    = ("status",)
    search_fields = ['title', 'content']
    prepopulated_fields = {'slug': ('title',)}

admin.site.register(Post, PostAdmin)
```

models.py

```
from django.db import models
from django.contrib.auth.models import User
```

```
STATUS = (
   (0,"Draft"),
   (1,"Publish")
)

class Post(models.Model):
   title = models.CharField(max_length=200, unique=True)
   slug = models.SlugField(max_length=200, unique=True)
   author = models.ForeignKey(User, on_delete=
   models.CASCADE,related_name='blog_posts')updated_on =
   models.DateTimeField(auto_now= True)
   content = models.TextField()
   created_on = models.DateTimeField(auto_now_add=True)
   status = models.IntegerField(choices=STATUS, default=0)

   class Meta:
      ordering = ['-created_on']

   def___str_(self):
      return self.title
```

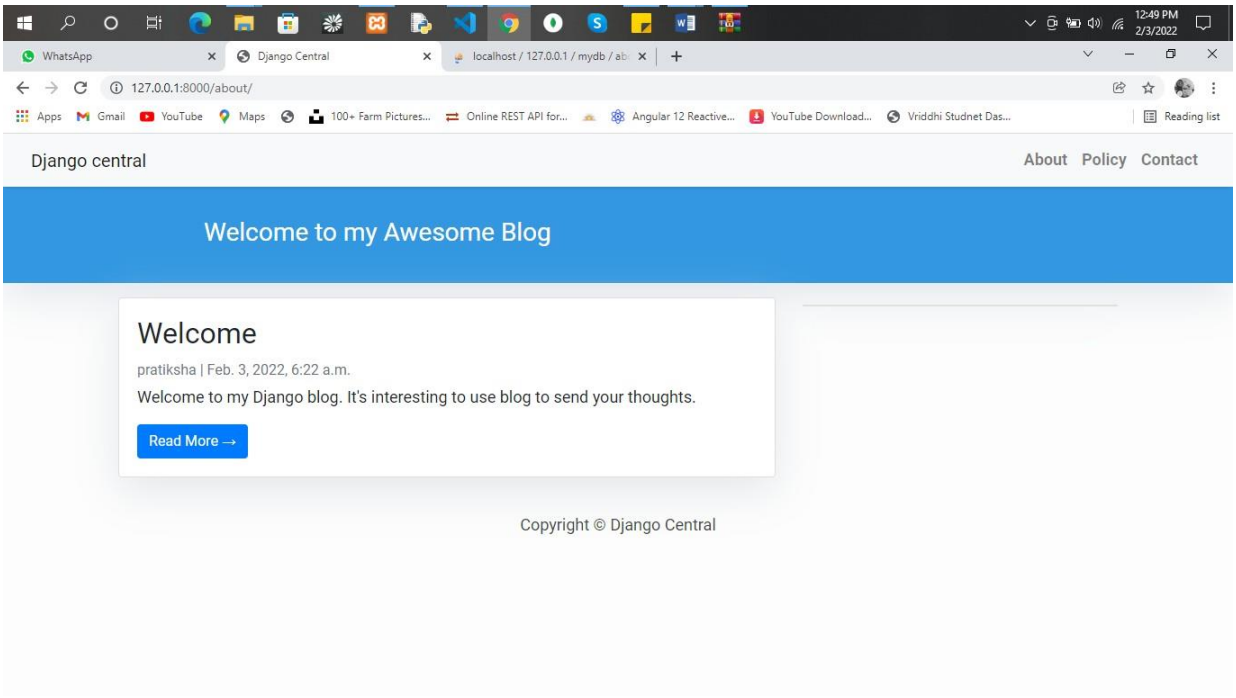
views.py

```
from django.views import generic
from .models import Post

class PostList(generic.ListView):
   queryset = Post.objects.filter(status=1).order_by('-created_on')
   template_name = 'index.html'

class
   PostDetail(generic.DetailView):
   model = Post
   template_name = 'post_detail.html'
```

urls.py

```
from . import views
from django.urls import path

urlpatterns = [
   path('', views.PostList.as_view(), name='about'),
   path('<slug:slug>/', views.PostDetail.as_view(), name='post_detail'),
]
```

Output:



Q. 1 Write a Java Program to implement State Pattern for Gumball Machine. Create instance variable that holds current state from there, we just need to handle all actions, behaviors andstate transition that can happen

1) Create Interface-

State.java

```
package javaprograms; public

interface State {

        public void insertQuarter();public
        void ejectQuarter(); public void
        turnCrank(); public void
        dispense();

        public void refill();
}
```

2) Create Class SoldState.java

```
package javaprograms;

public class SoldState implements State {

    GumballMachine gumballMachine;

    public SoldState(GumballMachine gumballMachine) {
        this.gumballMachine = gumballMachine;
    }
```

```
	public void insertQuarter() {
		System.out.println("Please wait, we're already giving you a
gumball");
	}

	public void ejectQuarter() {
		System.out.println("Sorry, you already turned the crank");
	}

	public void turnCrank() {
		System.out.println("Turning twice doesn't get you another
gumball!");
	}

	public void dispense() {
		gumballMachine.releaseBall();
		if (gumballMachine.getCount() > 0) {

	gumballMachine.setState(gumballMachine.getNoQuarterState());
		} else {
```

```
                        System.out.println("Oops, out of gumballs!");
                        gumballMachine.setState(gumballMachine.getSoldOutState());
                }
        }

        public void refill() { }

        public String toString() {
                return "dispensing a gumball";
        }
}
```

SoldOutState.java

```
package javaprograms;


public class SoldOutState implements State {
      GumballMachine gumballMachine;

      public SoldOutState(GumballMachine gumballMachine) {
            this.gumballMachine = gumballMachine;
      }

        public void insertQuarter() {
                System.out.println("You can't insert a quarter, the machine is
sold out");
        }

        public void ejectQuarter() {
                System.out.println("You can't eject, you haven't inserted aquarter yet");
        }

        public void turnCrank() {
                System.out.println("You turned, but there are no gumballs");
        }

        public void dispense() {
                System.out.println("No gumball dispensed");
        }

        public void refill() { gumballMachine.setState(gumballMachine.getNoQuarterState());
        }

        public String toString() {return
                "sold out";
        }
}
```

NoQuarterState.java

```
package javaprograms;

public class NoQuarterState implements State {
	GumballMachine gumballMachine;

	public NoQuarterState(GumballMachine gumballMachine) {
		this.gumballMachine = gumballMachine;
	}

	  public void insertQuarter() { System.out.println("You inserted a
		quarter");
		gumballMachine.setState(gumballMachine.getHasQuarterState());
	  }

	  public void ejectQuarter() {
			System.out.println("You haven't inserted a quarter");
	  }

	  public void turnCrank() {
		System.out.println("You turned, but there's no quarter");
	    }

	  public void dispense() {
		System.out.println("You need to pay first");
	  }

	  public void refill() { } public

	  String toString() {
		return "waiting for quarter";
	  }
}

HasQuarterState.java
package javaprograms;


public class HasQuarterState implements State {
	  GumballMachine gumballMachine;

	  public HasQuarterState(GumballMachine gumballMachine) {
		this.gumballMachine = gumballMachine;
	  }

	  public void insertQuarter() {
			System.out.println("You can't insert another quarter");
	  }

	  public void ejectQuarter() { System.out.println("Quarter
		returned");
```

```
                gumballMachine.setState(gumballMachine.getNoQuarterState());
        }

        public void turnCrank() { System.out.println("You
                turned...");
                gumballMachine.setState(gumballMachine.getSoldState());
        }

    public void dispense() {
        System.out.println("No gumball dispensed");
    }

    public void refill() { } public

        String toString() {
                return "waiting for turn of crank";
        }
}
```

GumballMachine.java

```
package javaprograms;


public class GumballMachine {

        State soldOutState; State
        noQuarterState; State
        hasQuarterState;State
        soldState;

        State state; int
        count = 0;

        public GumballMachine(int numberGumballs) { soldOutState
                = new SoldOutState(this); noQuarterState = new
                NoQuarterState(this); hasQuarterState = new
                HasQuarterState(this);soldState = new SoldState(this);

                this.count = numberGumballs;if
                (numberGumballs > 0) {
                        state = noQuarterState;
                } else {
                        state = soldOutState;
                }
        }

        public void insertQuarter() {
                state.insertQuarter();
        }
```

```
        public void ejectQuarter() {
                state.ejectQuarter();
        }

        public void turnCrank() {
                state.turnCrank();
                state.dispense();
        }

        void releaseBall() {
                System.out.println("A gumball comes rolling out the slot...");if (count > 0) {
                        count = count - 1;
                }
        }

        int getCount() {
                return count;
        }

        void refill(int count) { this.count
                += count;
                System.out.println("The gumball machine was just refilled; itsnew count is: "
+ this.count);
                state.refill();
        }

        void setState(State state) {this.state
                = state;
        }
    public State getState() {return
            state;
    }

    public State getSoldOutState() {return
            soldOutState;
    }

    public State getNoQuarterState() {return
            noQuarterState;
    }

    public State getHasQuarterState() {return
            hasQuarterState;
    }

    public State getSoldState() {return
            soldState;
    }
```

```
	public String toString() {
		StringBuffer result = new StringBuffer(); result.append("\nMighty Gumball,
		Inc."); result.append("\nJava-enabled Standing Gumball Model #2004");
		result.append("\nInventory: " + count + " gumball");
		if (count != 1) {
			result.append("s");
		}
		result.append("\n");
		result.append("Machine is " + state + "\n");return
		result.toString();
	}
}
```

GumballMachineTestDrive.javapackage
javaprograms;

```
public class GumballMachineTestDrive {

	public static void main(String[] args) {
		GumballMachine gumballMachine = new GumballMachine(2);

		System.out.println(gumballMachine);

		gumballMachine.insertQuarter();
		gumballMachine.turnCrank();

		System.out.println(gumballMachine);

		gumballMachine.insertQuarter();
		gumballMachine.turnCrank();
		gumballMachine.insertQuarter();
		gumballMachine.turnCrank();

		gumballMachine.refill(5);
		gumballMachine.insertQuarter();
		gumballMachine.turnCrank();

		System.out.println(gumballMachine);
	}
}
```

Output-
Mighty Gumball, Inc.
Java-enabled Standing Gumball Model #2004
Inventory: 2 gumballs
Machine is waiting for quarterYou

inserted a quarter

You turned...
A gumball comes rolling out the slot...

Mighty Gumball, Inc.
Java-enabled Standing Gumball Model #2004
Inventory: 1 gumball
Machine is waiting for quarter

You inserted a quarterYou
turned...
A gumball comes rolling out the slot...Oops, out of
gumballs!
You can't insert a quarter, the machine is sold outYou turned, but
there are no gumballs
No gumball dispensed
The gumball machine was just refilled; its new count is: 5You inserted a
quarter
You turned...
A gumball comes rolling out the slot...

Mighty Gumball, Inc.
Java-enabled Standing Gumball Model #2004
Inventory: 4 gumballs
Machine is waiting for quarter

Q.2. Write a python program to implement Decision Tree whether or not to play Tennis.

```
# Write a python program to Implement Decision Tree whether or not to play tennis.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

PlayTennis = pd.read_csv("PlayTennis.csv")

#We can convert all the non numerical values into numerical values usingLabelEncoder

from sklearn.preprocessing import LabelEncoderLe =
LabelEncoder()

PlayTennis['outlook'] = Le.fit_transform(PlayTennis['outlook']) PlayTennis['temp'] =
Le.fit_transform(PlayTennis['temp']) PlayTennis['humidity'] = Le.fit_transform(PlayTennis['humidity'])
PlayTennis['windy'] = Le.fit_transform(PlayTennis['windy']) PlayTennis['play'] =
Le.fit_transform(PlayTennis['play'])

#Lets split the training data and its coresponding prediction values.
```

```
#y - holds all the decisions.#X - holds
the training data.y = PlayTennis['play']
X = PlayTennis.drop(['play'],axis=1)

# Fitting the model
from sklearn import tree
clf = tree.DecisionTreeClassifier(criterion = 'entropy')   #A decision treeclassifier. “entropy” for the information
gain.
clf = clf.fit(X, y)                                                                    #Decision tree
algorithm splits nodes as long as this value decreases till it reaches zero

# We can visualize the tree using tree.plot_treetree.plot_tree(clf)
plt.show()
```

Q.1 Write a Java Program to implement Factory method for Pizza Store with createPizza(), orederPizza(), prepare(), Bake(), cut(), box(). Use this to create variety of pizza’s like NyStyleCheesePizza, ChicagoStyleCheesePizza etc.

Create Class –

1)Pizza.class

```
package javaprograms;
import java.util.ArrayList;

abstract public class Pizza {String
        name;
        String dough;
        String sauce;
        ArrayList toppings = new ArrayList();

        public String getName() {
                return name;
        }

        public void prepare() { System.out.println("Preparing " +
                name);
        }

        public void bake() { System.out.println("Baking " +
                name);
        }

        public void cut() { System.out.println("Cutting " +
                name);
        }

        public void box() { System.out.println("Boxing " +
                name);
        }

        public String toString() {
```

```
            // code to display pizza name and ingredientsStringBuffer
            display = new StringBuffer(); display.append("---- " + name
            + " ---------------------------------------------------- \n");
            display.append(dough + "\n");
            display.append(sauce + "\n");
            for (int i = 0; i < toppings.size(); i++) { display.append((String) toppings.get(i) + "\n");
            }
            return display.toString();
        }
}
```

2)PizzaStore.class

```
package javaprograms;

public class PizzaStore {
	SimplePizzaFactory factory;

	public PizzaStore(SimplePizzaFactory factory) {
		this.factory = factory;
	}

	public Pizza orderPizza(String type) {Pizza pizza;

		pizza = factory.createPizza(type);

		pizza.prepare();
		pizza.bake();
		pizza.cut();
		pizza.box();

		return pizza;
	}
}
```

3)SimplePizzaFactory.class

```
package javaprograms;

public class SimplePizzaFactory {

	public Pizza createPizza(String type) {Pizza pizza =
		null;

		if (type.equals("cheese")) { pizza = new
			NYCheesePizza();
		} else if (type.equals("veggie")) { pizza = new
			ChicagoCheesePizza();
		}
		return pizza;
	}
}
```

4)NYCheesePizza.class

```
package javaprograms;
public class NYCheesePizza extends Pizza {
	public   NYCheesePizza()   { name =
		"NY Cheese Pizza";dough =
		"Regular Crust";
		sauce = "Marinara Pizza Sauce";
```

```
            toppings.add("Fresh Mozzarella");
            toppings.add("Parmesan");
        }
}
```

5)ChicagoCheesePizza.class

```
package javaprograms;
public class ChicagoCheesePizza extends Pizza {
        public   ChicagoCheesePizza()   { name =
            "Chicago Cheese Pizza";dough =
            "Crust";
            sauce = "Marinara sauce";
            toppings.add("Shredded mozzarella");
            toppings.add("Grated parmesan");
            toppings.add("Diced onion");
            toppings.add("Sliced mushrooms");
            toppings.add("Sliced red pepper");
            toppings.add("Sliced black olives");
        }
}
```

Output-

Preparing NY Cheese Pizza
Baking NY Cheese Pizza Cutting
NY Cheese Pizza Boxing NY
Cheese Pizza
We ordered a NY Cheese Pizza

Preparing Chicago Cheese PizzaBaking
Chicago Cheese Pizza Cutting Chicago
Cheese Pizza Boxing Chicago Cheese
Pizza
We ordered a Chicago Cheese Pizza

Q.2. Write a python program to implement Linear SVM.

```
# Import the Librariesimport
numpy as np
import matplotlib.pyplot as plt from sklearn
import svm, datasets

# Import some Data from the iris Data Setiris =
datasets.load_iris()

# Take only the first two features of Data.
# To avoid the slicing, Two-Dim Dataset can be usedX = iris.data[:,

:2]
```

```
y = iris.target

# C is the SVM regularization parameterC = 1.0

# Create an Instance of SVM and Fit out the data.
# Data is not scaled so as to be able to plot the support vectors svc = svm.SVC(kernel
='linear', C = 1).fit(X, y) #Fit the SVM modelaccording to the given training data.
                                                                    #SVC=Support Vector
Classifier

# create a mesh to plot
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1h = (x_max /
x_min)/100
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),                    #Return
coordinate matrices from coordinate vectors.
            np.arange(y_min, y_max, h))

# Plot the data for Proper Visual Representation plt.subplot(1, 1, 1)
#Add a subplot to the currentfigure.subplot(nrows, ncols,
indexOfSubplot)

# Predict the result by giving Data to the model
Z = svc.predict(np.c_[xx.ravel(), yy.ravel()])                    #ravel()-Return a
contiguous flattened array.
Z = Z.reshape(xx.shape)
plt.contourf(xx, yy, Z, cmap = plt.cm.Paired, alpha = 0.8) #contourand contourf draw
contour lines and filled contours.

plt.scatter(X[:, 0], X[:, 1], c = y, cmap = plt.cm.Paired)plt.xlabel('Sepal length')
plt.ylabel('Sepal width') plt.xlim(xx.min(),
xx.max()) plt.title('SVC with linear kernel')

# Output the Plot
plt.show()
```

Q.3 Implement Login System using Django.

home app

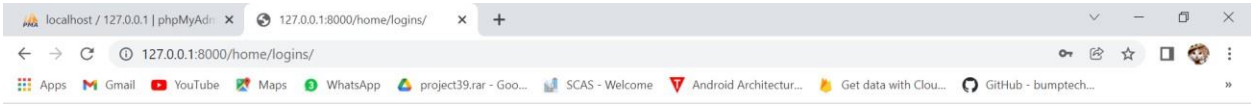
forms.py

```
from django import forms
```

```
class LoginForm(forms.Form):
        username=forms.CharField(max_length=63)
        password=forms.CharField(max_length=63,widget=forms.PasswordInput)

login.html
<form method="post">
        {{ form.as_p }}
        {% csrf_token %}
        <button type="submit">Submit</button>
</form>

views.py
from django.shortcuts import renderfrom .
import forms

def login(request):
        form=forms.LoginForm()
        if request.method=='POST':
                form=forms.LoginForm(request.POST)if
                form.is_valid():
                        return HttpResponse("Login Ok")
        return render(request,'login.html',context={'form':form})urls.py

from django.urls import pathfrom
.views import login urlpatterns = [
        path('logins/',login, name="home"),
]
test_project
from django.contrib import adminfrom
django.urls import path from django.urls
import include urlpatterns = [
        path('home/', include("home.urls")),
]
```


localhost / 127.0.0.1 | phpMyAdm
127.0.0.1:8000/home/logins/
127.0.0.1:8000/home/logins/
Apps
Gmail
YouTube
Maps
WhatsApp
project39.rar - Goo...
SCAS - Welcome
Android Architectur...
Get data with Clou...
GitHub - bumptech...
Username: kanchanghule2000@gmail.
Password:
Submit


Type here to search
19°C
8:10 PM
1/22/2022


localhost / 127.0.0.1 | phpMyAdm
127.0.0.1:8000/home/logins/
127.0.0.1:8000/home/logins/
Apps
Gmail
YouTube
Maps
WhatsApp
project39.rar - Goo...
SCAS - Welcome
Android Architectur...
Get data with Clou...
GitHub - bumptech...
Login Ok


Type here to search
19°C
8:10 PM
1/22/2022