**Script.sql**

drop database if exists RainfallReport;

create database RainfallReport

use RainfallReport;

create table AnnualRainfall(city\_pincode int(5) primary key,city\_name varchar(25),average\_annual\_rainfall double(6,2));

insert into AnnualRainfall values(10002,'New York',19.833);

insert into AnnualRainfall values(99501,'Alaska',22.666);

insert into AnnualRainfall values(20019,'Washington',35.006);

insert into AnnualRainfall values(30381,'Oxford',22.666);

insert into AnnualRainfall values(90080,'Angeles',42);

COMMIT;

**AnnualRainfall.java**

import java.util.\*;

public class AnnualRainfall {

private int cityPincode;

private String cityName;

private double averageAnnualRainfall;

public int getCityPincode() {

return cityPincode;

}

public void setCityPincode(int cityPincode) {

this.cityPincode = cityPincode;

}

public String getCityName(){

return cityName;

}

public void setCityName(String cityName){

this.cityName = cityName;

}

public double getAverageAnnualRainfall(){

return averageAnnualRainfall;

}

public void setAverageAnnualRainfall(double averageAnnualRainfall){

this.averageAnnualRainfall = averageAnnualRainfall;

}

//Write the required business logic as expected in the question description

public void calculateAverageAnnualRainfall (double monthlyRainfall [ ]){

//fill the code

double sum=0;

for(int i=0;i<12;i++)

{

sum=sum+monthlyRainfall[i];

}

sum/=12;

setAverageAnnualRainfall(sum);

}

}

**DbHandler.java**

import java.io.FileInputStream;

import java.io.IOException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.util.Properties;

public class DBHandler {

private static Connection con = null;

private static Properties props = new Properties();

//Write the required business logic as expected in the question description

public Connection establishConnection() throws ClassNotFoundException, SQLException {

try{

FileInputStream fis = null;

fis = new FileInputStream("db.properties");

props.load(fis);

// load the Driver Class

Class.forName(props.getProperty("db.classname"));

// create the connection now

con = DriverManager.getConnection(props.getProperty("db.url"),props.getProperty("db.username"),props.getProperty("db.password"));

}

catch(IOException e){

e.printStackTrace();

}

return con;

}

}

**InvalidException.java**

public class InvalidCityPincodeException extends Exception {

public InvalidCityPincodeException(String a) {

super(a);

}

}

**Main.Java**

import java.io.IOException;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

public class Main {

public static void main(String[] args)

throws IOException, InvalidCityPincodeException, ClassNotFoundException, SQLException {

RainfallReport rf = new RainfallReport();

List<AnnualRainfall> avgli = new ArrayList<AnnualRainfall>();

avgli = rf.generateRainfallReport("AllCityMonthlyRainfall.txt");

List<AnnualRainfall> maxli = new ArrayList<AnnualRainfall>();

maxli = rf.findMaximumRainfallCities();

for (int i = 0; i < maxli.size(); i++) {

AnnualRainfall ob = maxli.get(i);

System.out.println("City Pincode:" + ob.getCityPincode());

System.out.println("City Name:" + ob.getCityName());

System.out.println("Average RainFall:" + ob.getAverageAnnualRainfall());

}

}

}

**RainfallReport.java**

import java.util.\*;

import java.io.\*;

import java.sql.\*;

public class RainfallReport

{

//Write the required business logic as expected in the question description

public List<AnnualRainfall> generateRainfallReport(String filePath) {

//fill the code

try

{

File myObj = new File(filePath);

Scanner myReader = new Scanner(myObj);

ArrayList<AnnualRainfall> list=new ArrayList<>();

while (myReader.hasNextLine())

{

double monthlyRainfall[]=new double[12];

String data[] = myReader.nextLine().split(",");

try

{

if(validate(data[0]))

{

for(int i=0;i<12;i++)

{

monthlyRainfall[i]=Double.parseDouble(data[i+2]);

}

AnnualRainfall a=new AnnualRainfall();

a.setCityPincode(Integer.parseInt(data[0]));

a.setCityName(data[1]);

a.calculateAverageAnnualRainfall(monthlyRainfall);

list.add(a);

}

}

catch(InvalidCityPincodeException e)

{

}

}

myReader.close();

return list;

}

catch (FileNotFoundException e)

{

return null;

}

}

public List<AnnualRainfall> findMaximumRainfallCities() {

//fill the code

try

{

List<AnnualRainfall> list=new ArrayList<>();

DBHandler db=new DBHandler();

Connection con=db.establishConnection();

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery("select \* from AnnualRainfall where average\_annual\_rainfall = (select max(average\_annual\_rainfall) from AnnualRainfall)");

while(rs.next())

{

int cityPincode=rs.getInt(1);

String cityName=rs.getString(2);

double avgAnnualRainfall=rs.getDouble(3);

AnnualRainfall a=new AnnualRainfall();

a.setCityName(cityName);

a.setCityPincode(cityPincode);

a.setAverageAnnualRainfall(avgAnnualRainfall);

list.add(a);

}

return list;

}

catch(Exception e)

{

return null;

}

}

public boolean validate(String cityPincode) throws InvalidCityPincodeException {

//fill the code

if(cityPincode.length()==5)

{

for(int i=0;i<5;i++)

{

char ch=cityPincode.charAt(i);

if(!(Character.isDigit(ch)))

{

throw new InvalidCityPincodeException("Invalid City Pincode");

}

}

return true;

}

throw new InvalidCityPincodeException("Invalid City Pincode");

}

}

**Dbproperties**

db.classname=com.mysql.jdbc.Driver

db.url=jdbc:mysql://localhost:3306/RainfallReport

db.username=

db.password=

**AllCityMonthlyRainfall.txt**

10002,New York,24,15,17,16,11,10,9,4,24,36,40,32

99501,Alaska,14,24,36,40,27,16,15,34,15,10,9,32

20019,Washington,11,20,12,20,10,15,20,21,19,10,11,12

30381,Atlanta,25,26,23,20,21,22,21,21,20,19,18,11

27565,Oxford,6,7,11,14,9,8,14,15,9,9,10,11

85054,Phoenix,12,11,12,11,10,14,15,14,15,12,10,9

90080,Los Angeles,6,5,4,5,4,5,6,8,4,5,9,4