AnnaulRainfall.java

```
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;
	}


	public void calculateAverageAnnualRainfall (double monthlyRainfall [ ]){

		double average=0;
	   for(int i=0;i<monthlyRainfall.length;i++)
	   {
		average+=monthlyRainfall[i];
	   }
	   average/=12;
	   this.averageAnnualRainfall=average;
```

```
	}



}
```

DBHanlder.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 {


		public Connection establishConnection() throws IOException, SQLException,
ClassNotFoundException {
			Properties p=new Properties();
			FileInputStream f=new FileInputStream("db.properties");
			p.load(f);
			Class.forName(p.getProperty("db.classname"));
			String url=p.getProperty("db.url");
			String username=p.getProperty("db.username");
			String password=p.getProperty("db.password");
			Connection c=DriverManager.getConnection(url,username,password);
			return c;
	}
}
```

InvalidCityPincode.java

```
@SuppressWarnings("serial")
public class InvalidCityPincodeException extends Exception {
			public InvalidCityPincodeException(String s)
	{
			super(s);
	}
}
```

Main.java

```
import java.io.IOException;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

public class Main {

   public static void main(String[] args) throws IOException, SQLException,
ClassNotFoundException {

            List<AnnualRainfall> li = new ArrayList<>();
                  RainfallReport r = new RainfallReport();
                  li = r.generateRainfallReport("AllCityMonthlyRainfall.txt");
                  Connection con = null;
                  Statement st = null;
                  DBHandler d=new DBHandler();
                  con = d.establishConnection();
                  st = con.createStatement();

                  for (int i = 0; i < li.size(); i++) {
         String sql = "INSERT INTO ANNUALRAINFALL VALUES(" + li.get(i).getCityPincode() +
"," + "'" + li.get(i).getCityName() + "'" + "," + li.get(i).getAverageAnnualRainfall() + ")" + "on
duplicate key update city_name=" + "'" + li.get(i).getCityName() + "'" + "," +
"average_annual_rainfall = " + li.get(i).getAverageAnnualRainfall() + ";" ;
         st.executeUpdate(sql);
      }
                  List<AnnualRainfall> finalList=new ArrayList<>();
                  finalList = r.findMaximumRainfallCities();
                  for (int i = 0; i < finalList.size(); i++) {
                        System.out.println(finalList.get(i).getCityName());
                  }

   }
}
```

Rainfallreport.java

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

public class RainfallReport {


	public List<AnnualRainfall> generateRainfallReport(String filePath) throws IOException {

		List<AnnualRainfall> avgList=new ArrayList<>();
		FileReader fr = new FileReader(new File(filePath));
		BufferedReader br = new BufferedReader(fr);
		String l;
		while((l=br.readLine())!=null)
		{
			String[] a=l.split(",");
			String pincode=a[0];
			try
			{
				if(validate(pincode))
				{
					double[] monthlyRainFall=new double[12];
					for(int i=2;i<=13;i++)
					{
						monthlyRainFall[i-2]=Double.parseDouble(a[i]);
					}
					AnnualRainfall ar=new AnnualRainfall();
					ar.calculateAverageAnnualRainfall(monthlyRainFall);
					ar.setCityName(a[1]);
					ar.setCityPincode(Integer.parseInt(pincode));
					avgList.add(ar);
				}
			}
			catch(InvalidCityPincodeException e)
			{
```

```
                        System.out.println(e.getMessage());
                    }
                }
                br.close();
                return avgList;
        }

        public List<AnnualRainfall>  findMaximumRainfallCities() throws SQLException,
ClassNotFoundException, IOException  {
                DBHandler d=new DBHandler();
                List<AnnualRainfall> finalList=new ArrayList<>();
                Connection c=d.establishConnection();
                Statement s=c.createStatement();
                String sql = "SELECT * FROM ANNUALRAINFALL WHERE
AVERAGE_ANNUAL_RAINFALL IN (SELECT MAX(AVERAGE_ANNUAL_RAINFALL) FROM
ANNUALRAINFALL)";
                ResultSet rs=s.executeQuery(sql);
                while(rs.next())
                {
                        AnnualRainfall ar=new AnnualRainfall();
                        ar.setCityName(rs.getString(2));
                        ar.setCityPincode(Integer.parseInt(rs.getString(1)));
                        ar.setAverageAnnualRainfall(Double.parseDouble(rs.getString(3)));
                        finalList.add(ar);
                }
                return finalList;
        }


        public boolean validate(String cityPincode) throws InvalidCityPincodeException {
                if(cityPincode.length()==5)
                {
                        return true;
                }
                else
                {
                        throw new InvalidCityPincodeException("Invalid CityPincode Exception");
                }
        }
}
```