```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package practica01;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author Richard Vivanco
 */
public class Coordenada {
    public float latitud;
    public float longitud;
    public String ciudad;


    public Coordenada() {
        latitud  =-2.1951f;
        longitud =-71.88f;
        ciudad="Guayaquil";
    }

    public Coordenada(float latitud, float longitud, String ciudad) {
        this.latitud = latitud;
        this.longitud = longitud;
        this.ciudad=ciudad;

    }

    @Override
    public String toString() {
        return ciudad+"("+latitud+","+longitud+")";
    }

    @Override
    public int hashCode() {
        int hash = 7;
        hash = 59 * hash + Float.floatToIntBits(this.latitud);
        hash = 59 * hash + Float.floatToIntBits(this.longitud);
        return hash;
    }

    @Override
    public boolean equals(Object obj) {
        if (obj == null) {
            return false;
        }
        if (getClass() != obj.getClass()) {
            return false;
        }
        Coordenada c = (Coordenada) obj;
        if(c.latitud==this.latitud && c.longitud==this.longitud)
            return true;
        return false;
```

```
    }
    /**
        * Retorna la distancia en Km entre esta instancia(this) y la
        * que recibe por p⌂rametro.
        * @param c reprecenta la coordenada con la que se calcular⌂ la distancia

        * @return Retorna la distancia entra la coordenada actual y la
coordenada
    */


    public double calcularDistancia(Coordenada c){
        int R=6373;

        double lat1rad=Math.toRadians(this.latitud);
        double lat2rad=Math.toRadians(c.latitud);
        double deltaLat= Math.toRadians(c.latitud-this.latitud);
        double deltaLon= Math.toRadians(c.longitud-this.longitud);

        double a =Math.sin(deltaLat/2)*
Math.sin(deltaLat/2)+Math.cos(lat1rad)*Math.cos(lat2rad)*
                Math.sin(deltaLon/2)*Math.sin(deltaLon/2);

        double x=2*Math.atan2(Math.sqrt(a), Math.sqrt(1-a));

        double d=R*x;

        return d;
    }

    public static LinkedList<Coordenada> cargar (String archivo){
        FileReader fr;
        BufferedReader br=null;
        String linea;
        String valores[];
        Coordenada c;
        LinkedList<Coordenada> lista=new LinkedList<Coordenada>();
        try {
            fr=new FileReader(archivo);
            br=new BufferedReader(fr);
            while((linea= br.readLine())!=null){
                valores =linea.split(",");
                c=new Coordenada ();
                c.ciudad=valores[2];
                float x=Float.parseFloat(valores[0]);
                float y=Float.parseFloat(valores[1]);
                c.latitud=x;
                c.longitud=y;
                lista.add(c);
            }
        } catch (FileNotFoundException ex) {
            System.out.println("El archivo no existe");
        } catch (IOException ex) {
            System.out.println("Error de Lectura");
        }

        return lista;

    }

    public void List_RemoveLastOcurrence (LinkedList<Coordenada> lista,
Coordenada  numero){
       Coordenada tmp, lastOcurrence=null;
       Iterator<Coordenada> i;
```

```
        i= lista.iterator();
        while(i.hasNext()){
            tmp=i.next();
            if(tmp.equals(numero)){
                lastOcurrence=tmp;
            }
        }
         if(lastOcurrence !=null){
             lista.remove(lastOcurrence);
         }
    }
}
```