```
/*
 * 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.
 */

/**
 *
 * @author gamer
 */
import java.io.*;
import java.math.BigInteger;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.io.FileNotFoundException;
import java.util.*;
import java.io.IOException;

class Main {

   /*Generacion de metodos para experimento
  *
  *
    */
   public static long regresaTiempo(HashTable<String> tabla, String nombre, String dato) {
      long startTime = System.nanoTime();
      long stopTime = 0;
      try {
```

```
            Double pos = fnHash(dato);
            tabla.add(dato, pos);

            if (nombre.equals("add")) {
                tabla.add(dato, pos);
                stopTime = System.nanoTime();
            } else if (nombre.equals("delete")) {
                tabla.delete(dato, pos);
                stopTime = System.nanoTime();
            } else if (nombre.equals("find")) {
                tabla.find(dato, pos);
                stopTime = System.nanoTime();
            }
        } catch (Exception e) {

        }
        return stopTime - startTime;
    }

    public static void llenaTabla(HashTable<String> table, String[] names) {
        for (int i = 0; i < names.length; i++) {
            String elem = names[i];
            try {
                Double pos = fnHash(elem);
                table.add(elem, pos);
            } catch (Exception e) {

            }
        }
```

```
}

public static byte[] getSHA(String input) throws NoSuchAlgorithmException {
    MessageDigest md = MessageDigest.getInstance("SHA-256");
    return md.digest(input.getBytes(StandardCharsets.UTF_8));
}

public static String toHexString(byte[] hash) {
    BigInteger numero = new BigInteger(1, hash);
    StringBuilder hexString = new StringBuilder(numero.toString(16));

    // es necesario rellenar con ceros
    while (hexString.length() < 32) {
        hexString.insert(0, '0');
    }
    return hexString.toString();
}

public static Double fnHash(String s1) throws NoSuchAlgorithmException {
    char[] auxCad = toHexString(getSHA(s1)).toCharArray();
    String res;
    int j = 0;
    Double num = 0.0;
    for (int i = auxCad.length - 1; i >= 0; i--) {
        res = "" + auxCad[i];
        num += Integer.parseUnsignedInt(res, 16) * Math.pow(16, j);
        j++;
    }
    return num;
```

```
    }

    public static String randomChoice(String[][] mat) {
        Random rand = new Random();
        int pos = rand.nextInt(mat.length);
        return mat[pos][0];
    }

    public static <T> void createCsv(T[][] mat, String nombre) throws IOException {
        FileWriter csvWriter = new FileWriter(nombre + ".csv");
        for (int i = 0; i < mat.length; i++) {
            for (int j = 0; j < mat[i].length; j++) {
                //csvWriter.write(String.join(",",mat[i][j]));
                if(j==mat[i].length-1)
                  csvWriter.write(mat[i][j]+"\n");
                else
                  csvWriter.write(mat[i][j]+", ");
            }
        }
        csvWriter.flush();
        csvWriter.close();
    }

    public static void main(String[] args) throws FileNotFoundException, NoSuchAlgorithmException
{
        Scanner sc = new Scanner(new File("Athletes.csv"));
        String[][] datos = new String[11100][3];
        int rows = 0;
        String str;
```

```
String[] aux;
while (sc.hasNextLine()) {
   str = sc.nextLine();
   aux = str.split(",");
   for (int i = 0; i < 3; i++) {
      datos[rows][i] = aux[i];
   }
   rows++;
}
sc.close();
/*
for (int i = 0; i < rows; i++) {
   System.out.print(datos[i][0] + "\t\t\t\t\t\t");
   System.out.print(datos[i][1] + "\t\t\t\t\t\t");
   System.out.print(datos[i][2] + "\n");
}
//prueba funcion hash
String s1 = "Guillermo Arredondo Renero";
double res = fnHash(s1);
System.out.println(res + " : " + (int) res);
System.out.println((int) (res % 9000));*/

//Experimento:2000, 10000, 11000, 12000, 18000, 25000, 30000
int[] tamanios = {500, 1000, 2000, 10000, 11000, 12000, 25000, 30000};

String atleta = "";
HashTable<String> atlet;
//for de tamanios --> for de operaciones con switch --> for de 30 veces
Double[][]inserta = new Double[30][tamanios.length+1];
```

```
Double[][]Busca = new Double[30][tamanios.length+1];
Double[][]Elimina = new Double[30][tamanios.length+1];
Double[][]Colisiones = new Double[30][tamanios.length+1];
for (int tam = 0; tam < tamanios.length; tam++) {

   for (int vez = 0; vez < 30; vez++) {
      atlet = new HashTable<String>(tamanios[tam]);

      double promedioInserta=0;
      double promedioBusca = 0;
      double promedioElimina = 0;
      double promedioCol = 0;
      long tiempo=0;
      for (int i = 0; i < datos.length; i++) {
         atleta = datos[i][0];
         tiempo = regresaTiempo(atlet, "add", atleta);
         double t = (double) tiempo;
         if(t>=0)
           promedioInserta=promedioInserta+t;
      }
      promedioInserta=promedioInserta/datos.length;
      inserta[vez][0] = (double)vez;
      inserta[vez][tam+1] = promedioInserta;
      promedioCol = atlet.promColisiones();

      long tiempoBusca =0;
      long tiempoElimina=0;
      for (int j = 0;j<3;j++){
        String dato = randomChoice(datos);
```

```
            tiempoBusca = regresaTiempo(atlet,"find",dato);
            tiempoElimina=regresaTiempo(atlet,"delete",dato);
            if(tiempoBusca>=0)
              promedioBusca = promedioBusca+(double)tiempoBusca;
            if(tiempoElimina>=0)
              promedioElimina = promedioElimina+(double)tiempoElimina;
          }
          promedioElimina=promedioElimina/datos.length;
          promedioBusca = promedioBusca/datos.length;
          promedioCol = promedioCol / datos.length;
          Busca[vez][0] = (double)vez;
          Busca[vez][tam+1] = promedioBusca;
          Elimina[vez][0] = (double)vez;
          Elimina[vez][tam+1] = promedioElimina;
          Colisiones[vez][0] = (double)vez;
          Colisiones[vez][tam+1] = promedioCol;

        }
      }

      try {
        createCsv(inserta, "Inserta");
        createCsv(Busca, "Busca");
        createCsv(Colisiones, "Colisiones");
        createCsv(Elimina, "Elimina");
      } catch (Exception e) {

      }
    }
```

```
}
```