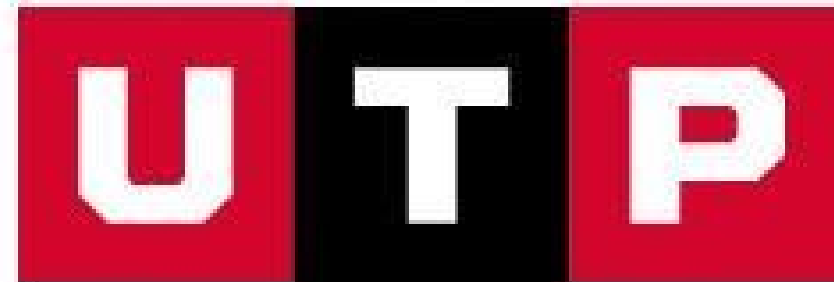


ANALISIS Y DISEÑO DE ALGORITMOS

Sesión 14: Algoritmos de Búsqueda Interna: TABLAS HASH:
Realización de una tabla dispersa. Direcccionamiento enlazado.



Universidad
Tecnológica
del Perú

Inicio

¿Tienen alguna consulta o duda sobre la clase previa?

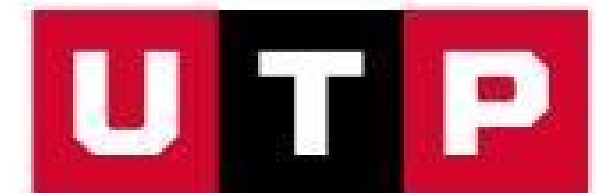


Logro de la sesión

Al finalizar la sesión, el estudiante aplica los aplica los métodos algorítmicos de búsqueda para la resolución de problemas.



PRACTICA



Universidad
Tecnológica
del Perú

IMPLEMENTACIÓN DE TABLAS HASH:

Exploración de Direcciones

1. En java implemente una tabla Hash el cual cuente con las operaciones básicas de insertar elemento y mostrar, además cuente con la función Hash Aritmética modular y evite colisión utilizando la exploración lineal, como claves debe ingresarse por código 6 elementos (100, “Juan”; 102, “Jorge”; 303, “María”, 132, “Juana”, 133, “Johana”, 134, “Julio”) y tamaño de arreglo 11.
2. Al enunciado anterior adicionar las operaciones básicas eliminar y buscar.

IMPLEMENTACIÓN DE TABLAS HASH:

Exploración de Direcciones (Solución)

```
Source History [Icons]
5 package ExploracionLineal;
6
7 /**
8  *
9  * @author Usuario
10 */
11 public class TablaHash {
12     private Entry[] table;
13     private int size;
14
15     public TablaHash(int size) {
16         this.size = size;
17         table = new Entry[size];
18     }
19
20     private int FuncionAritmeticaModular(int clave) {
21         return clave % size; // Función aritmética modular
22     }
23
24     public void ingresar(int clave, String value) {
25         int hash = FuncionAritmeticaModular(clave);
26         while (table[hash] != null) {
27             hash = (hash + 1) % size; // Exploración lineal
28         }
29         table[hash] = new Entry(clave, value);
30     }
}
```

IMPLEMENTACIÓN DE TABLAS HASH:

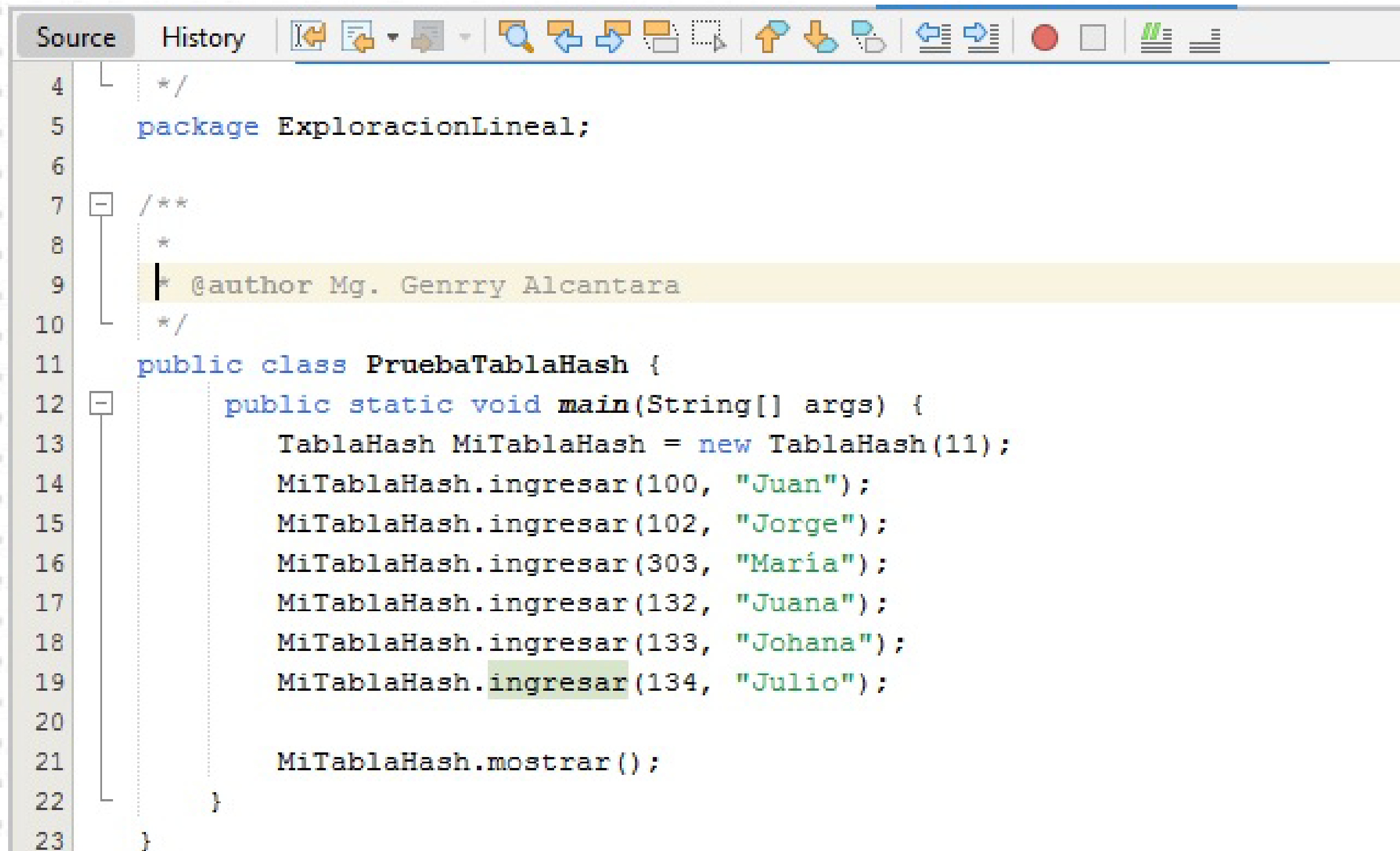
Exploración de Direcciones (Solución)

```
Source History [Icons]
31
32 public void mostrar() {
33     for (int i = 0; i < size; i++) {
34         if (table[i] != null) {
35             System.out.println("Posición " + i + ": " + table[i].clave + " -> " + table[i].value);
36         } else {
37             System.out.println("Posición " + i + ": vacía");
38         }
39     }
40 }

69 private static class Entry {
70     int clave;
71     String value;
72
73     Entry(int clave, String value) {
74         this.clave = clave;
75         this.value = value;
76     }
77 }
78 }
```

IMPLEMENTACIÓN DE TABLAS HASH:

Exploración de Direcciones (Solución)



The image shows a screenshot of an IDE with a toolbar at the top. The toolbar includes icons for Source, History, and various editing and navigation functions. The code is written in Java and is as follows:

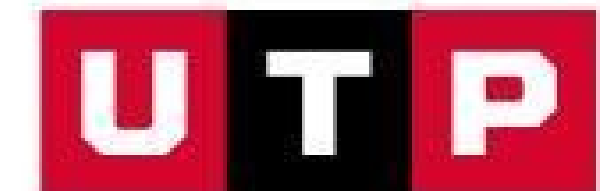
```
4  */
5  package ExploracionLineal;
6
7  /**
8   *
9   * @author Mg. Genrry Alcantara
10  */
11  public class PruebaTablaHash {
12      public static void main(String[] args) {
13          TablaHash MiTablaHash = new TablaHash(11);
14          MiTablaHash.ingresar(100, "Juan");
15          MiTablaHash.ingresar(102, "Jorge");
16          MiTablaHash.ingresar(303, "María");
17          MiTablaHash.ingresar(132, "Juana");
18          MiTablaHash.ingresar(133, "Johana");
19          MiTablaHash.ingresar(134, "Julio");
20
21          MiTablaHash.mostrar();
22      }
23  }
```


IMPLEMENTACIÓN DE TABLAS HASH:

Exploración de Direcciones (Solución)

Output - Run (PruebaTablaHash)

```
-----[ jar ]-----  
--- exec-maven-plugin:3.0.0:exec (default-cli) @ EjemploSesion ---  
Posición 0: 132 -> Juana  
Posición 1: 100 -> Juan  
Posición 2: 133 -> Johana  
Posición 3: 102 -> Jorge  
Posición 4: 134 -> Julio  
Posición 5: vacía  
Posición 6: 303 -> María  
Posición 7: vacía  
Posición 8: vacía  
Posición 9: vacía  
Posición 10: vacía  
-----  
BUILD SUCCESS
```



Universidad
Tecnológica
del Perú

IMPLEMENTACIÓN DE TABLAS HASH:

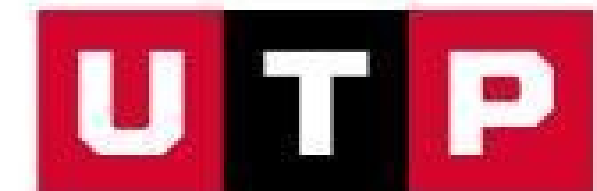
Exploración de Direcciones

2. Al enunciado anterior adicionar las operaciones básicas eliminar y buscar.

IMPLEMENTACIÓN DE TABLAS HASH:

Exploración de Direcciones (Solución)

```
Source History [Icons]
42
43 [-] public void eliminar(int clave) {
44     int hash = FuncionAritmeticaModular(clave);
45     while (table[hash] != null) {
46         if (table[hash].clave == clave) {
47             table[hash] = null;
48             System.out.println("Elemento con clave " + clave + " eliminado.");
49             return;
50         }
51         hash = (hash + 1) % size; // Exploración lineal
52     }
53     System.out.println("Elemento con clave " + clave + " no encontrado.");
54 }
55
56 [-] public void buscar(int clave) {
57     int hash = FuncionAritmeticaModular(clave);
58     while (table[hash] != null) {
59         if (table[hash].clave == clave) {
60             //return table[hash].value;
61             System.out.println("Elemento: " + table[hash].value);
62             return;
63         }
64         hash = (hash + 1) % size; // Exploración lineal
65     }
66     System.out.println("Elemento con clave " + clave + " no encontrado.");
67 }
68
```



Universidad
Tecnológica
del Perú

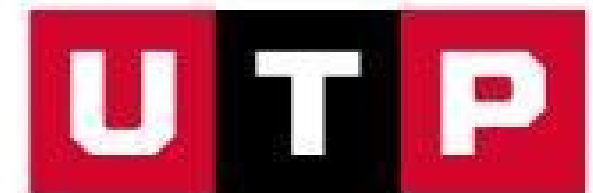
IMPLEMENTACIÓN DE TABLAS HASH:

Exploración de Direcciones (Solución)

```
Source History [Icons]
4  */
5  package ExploracionLineal;
6
7  /**
8   *
9   * @author Mg. Genrry Alcantara
10  */
11  public class PruebaTablaHash {
12      public static void main(String[] args) {
13          TablaHash MiTablaHash = new TablaHash(11);
14          MiTablaHash.ingresar(100, "Juan");
15          MiTablaHash.ingresar(102, "Jorge");
16          MiTablaHash.ingresar(303, "María");
17          MiTablaHash.ingresar(132, "Juana");
18          MiTablaHash.ingresar(133, "Johana");
19          MiTablaHash.ingresar(134, "Julio");
20
21          MiTablaHash.mostrar();
22
23          MiTablaHash.buscar(134);
24          MiTablaHash.eliminar(134);
25          MiTablaHash.mostrar();
26      }
27  }
```

EJERCICIOS

1. Del enunciado anterior implemente utilizando la exploración cuadrática.
2. Implemente un programa en java (en base a lo anterior) que solicite las claves y Nombres de un grupo de estudiantes de un instituto y estos se inserten en una tabla Hash utilizando el método de dispersión **por plegamiento**, estos deben utilizar la función de exploración cuadrática para evitar la colisión de índices; los datos deben ser ingresados por el usuario, el cual al finalizar de ingresarlos, el programa debe mostrar la tabla como resultado. Las claves son de 9 dígitos (ejemplos: 100010001, 109100023, 207200301, etc.) y su tamaño de arreglo de 12.



¿Tienen alguna consulta o duda?

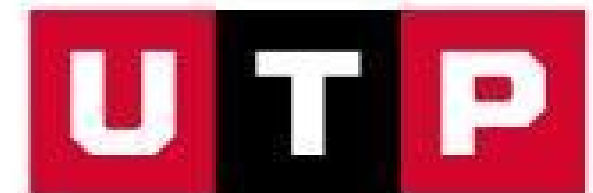


Cierre



¿ Con qué aprendizaje nos vamos?

Elaboramos nuestras conclusiones sobre el tema tratado



Universidad
Tecnológica
del Perú



**Universidad
Tecnológica
del Perú**