

## Exámenes

### Tema 3 - S3: Cuestiones sobre Hashing II

[Volver a la Lista de Exámenes](#)

Parte 1 de 2 - Cuestiones sobre Colisiones, Factor de Carga, Rehashing y Tamaño de la Tabla

5.5/ 5.5 Puntos

Preguntas 1 de 8

1.0/ 1.0 Puntos

Selecciona de entre las siguientes frases aquellas (1 o más) que sean ciertas.

- ☒ Dos claves `c1` y `c2` tales que `c1.equals(c2)` tienen el mismo índice *Hash*.
- ☐ Dos claves `c1` y `c2` tales que `!c1.equals(c2)` tienen distinto índice *Hash*.
- ☐ Para dos claves `c1` y `c2` que tienen el mismo índice *Hash* se cumple necesariamente que `c1.equals(c2)`.
- ☒ Para dos claves `c1` y `c2` que tienen distinto índice *Hash* se cumple necesariamente que `!c1.equals(c2)`.

Respuesta correcta: A, D

Preguntas 2 de 8

1.5/ 1.5 Puntos

Al usar el método `hashCode` "Malo", que figura a continuación, como Función de Dispersión de una Tabla Hash de talla 109580 y claves de tipo `String` con longitud máxima 28, el número de colisiones que se producen es constante, tanto si el tamaño de la tabla (`elArray.length`) es igual a su talla como si es hasta 10 veces mayor que esta; precisamente por ello, el comportamiento de la Tabla Hash resultante es muy "Malo".

```
public int hashCode() {  
    int valorHash = 0;  
  
    for (int i = 0; i < clave.length(); i++) { valorHash += clave.charAt(i); }  
  
    return valorHash;  
}
```

Para explicar este mal comportamiento basta determinar el rango de posiciones de `elArray` que permite indexar el método "Malo" y observar que es mucho menor que la talla de la Tabla (109580)... ¿Cuál de los siguientes es dicho rango?

**PISTA:** el resultado de `clave.charAt(i)` es un valor en el intervalo [0, 127]

- ☐ [3556, 109579], pues la talla de la Tabla es 109580 y 3556 es el valor máximo que puede devolver el método "Malo" (clave compuesta de 28 caracteres iguales y con `charAt(i) = 127`).
- ☒ [0, 3556]  
Comentarios:  
El valor mínimo que puede devolver el método `hashCode` "Malo" es 0 (clave de longitud 0) y el máximo  $28 * 127 = 3556$  (clave compuesta de 28 caracteres iguales y con `charAt(i) = 127`). Por tanto, este método:
  - solo permite indexar las 3557 primeras posiciones de `elArray` ( $3556 - 0 + 1$ );
  - como la talla de la Tabla es 109580, provoca en el Mejor de los Casos, como mínimo,  $109580 - 3557$  colisiones; nota que este número mínimo de colisiones es el mismo tanto si el tamaño de la tabla (`elArray.length`) es igual a su talla como si es hasta 10 veces mayor que esta.
- ☐ [0, 109579], pues la talla de la Tabla es 109580.

Respuesta correcta:B

Se quiere representar una Colección de 100 Entradas mediante una Tabla *Hash* Enlazada con  $FC = 0.75$  ¿Cuál es la mejor capacidad, o la mejor longitud de `elArray`, que debe tener dicha tabla en este caso?

- 
- ☐ 100
  - ☐ 101
  - ☐ 211
  - ☐ 133
  - ☒ 137

Comentarios:

Si bien `elArray.length` = talla máxima estimada / FC, por lo que para este caso `elArray.length` sería 133, se debe recordar que la elección de un  $n^\circ$  primo favorece la dispersión. Por tanto, `siguientePrimo(133) = 137` es la mejor capacidad de la Tabla *Hash*.

Respuesta correcta:E

## Preguntas 4 de 8

1.0/ 1.0 Puntos

Se tiene una Tabla *Hash* Enlazada vacía de capacidad 5, claves de tipo *Integer* y, por tanto, función de dispersión

$\text{indiceHash}(c) = c \% 5$ ; sin hacer *Rehashing*, se han insertado en dicha tabla los siguientes *Integer*: 9, 7, 3, 17, 18, 16, 12, 10, 22.

¿Cuál es la Tabla *Hash* resultante? Dibújala en un papel y, luego, ....

(a) Indica (con una cifra decimal) cuál es su Factor de Carga: FC = ✓ 1.8

(b) Completa el siguiente cuadro, cada una de cuyas filas representa una cubeta de la Tabla *Hash* que has dibujado.

Para ello debes escribir en cada fila  $i$  del cuadro las claves que almacena la cubeta  $i$  de la Tabla, rellenando con el símbolo - (guión normal) las celdas de la fila que sean innecesarias. Así, por ejemplo, si en la cubeta 0 sólo estuviera la clave 10, deberías escribir

|   |    |   |   |   |
|---|----|---|---|---|
| 0 | 10 | - | - | - |
|---|----|---|---|---|

| Fila/Cubeta |             |             |             |             |
|-------------|-------------|-------------|-------------|-------------|
| 0           | ✓ <u>10</u> | ✓ -         | ✓ -         | ✓ -         |
| 1           | ✓ <u>16</u> | ✓ -         | ✓ -         | ✓ -         |
| 2           | ✓ <u>7</u>  | ✓ <u>17</u> | ✓ <u>12</u> | ✓ <u>22</u> |
| 3           | ✓ <u>3</u>  | ✓ <u>18</u> | ✓ -         | ✓ -         |
| 4           | ✓ <u>9</u>  | ✓ -         | ✓ -         | ✓ -         |

Respuesta correcta: 1.8|1,8, 10, -, -, -, 16, -, -, -, 7|22, 17|12, 12|17, 22|7, 3|18, 18|3, -, -, 9, -, -, -

## Preguntas 5 de 8

1.0/ 1.0 Puntos

En una Tabla *Hash* Enlazada de capacidad 5, claves de tipo *Integer* y función de dispersión  $\text{indiceHash}(c) = c \% 5$

se han insertado los siguientes *Integer*: 9, 7, 3, 17, 18, 16, 12, 10, 22.

Teniendo en cuenta que tras insertar una de las claves en la Tabla su Factor de Carga ha superado el valor 0.75 y, por tanto, se ha tenido que efectuar un *Rehashing* que ha hecho aumentar su capacidad a 11, dibuja en un papel la Tabla *Hash* resultante tras la inserción del último elemento (el 22) y, luego, ...

(a) Indica (con dos cifras decimales) cuál es el Factor de Carga de la Tabla tras insertar el 22: FC = ✓ 0.82

(b) Completa el siguiente cuadro, cada una de cuyas filas representa una cubeta de la Tabla *Hash* que has dibujado.

Para ello debes escribir en cada fila  $i$  del cuadro las claves que almacena la cubeta  $i$  de la Tabla, rellenando con el símbolo - (guión normal) las celdas de la fila que sean innecesarias. Así, por ejemplo, si en la cubeta 0 sólo estuviera la clave 10, deberías escribir

|   |    |   |   |   |
|---|----|---|---|---|
| 0 | 10 | - | - | - |
|---|----|---|---|---|

| Fila/Cubeta |             |             |     |     |
|-------------|-------------|-------------|-----|-----|
| 0           | ✓ <u>22</u> | ✓ =         | ✓ = | ✓ = |
| 1           | ✓ <u>12</u> | ✓ =         | ✓ = | ✓ = |
| 2           | ✓ =         | ✓ =         | ✓ = | ✓ = |
| 3           | ✓ <u>3</u>  | ✓ =         | ✓ = | ✓ = |
| 4           | ✓ =         | ✓ =         | ✓ = | ✓ = |
| 5           | ✓ <u>16</u> | ✓ =         | ✓ = | ✓ = |
| 6           | ✓ <u>17</u> | ✓ =         | ✓ = | ✓ = |
| 7           | ✓ <u>7</u>  | ✓ <u>18</u> | ✓ = | ✓ = |
| 8           | ✓ =         | ✓ =         | ✓ = | ✓ = |
| 9           | ✓ <u>9</u>  | ✓ =         | ✓ = | ✓ = |
| 10          | ✓ <u>10</u> | ✓ =         | ✓ = | ✓ = |

**Respuesta correcta:**0.82|0,82, 22, -, -, 12, -, -, -, -, -, 3, -, -, -, -, -, 16, -, -, 17, -, -, 7|18, 18|7, -, -, -, -, -, 9, -, -, 10, -, -, -

#### Comentarios:

La capacidad inicial de la Tabla (`elArray.length`) es 5. PERO, tras insertar en ella las claves 9, 7, 3 y 17 ...

- 0 --
- 1 --
- 2 -- 7, 17
- 3 -- 3
- 4 -- 9

el Factor de Carga vale 0.8 ( $FC > 0.75$ ) y se efectúa un *Rehashing*. Por tanto, la **capacidad de la Tabla pasa a ser 11** y las claves se **redispersan con la función  $c \% 11$** . Así, tras insertar la última clave (el 22) la Tabla queda en el estado que se indica en la solución de este ejercicio.

**Date cuenta que justo tras insertar la última de las claves (el 22) el Factor de Carga pasa a ser 0.82 ( $FC > 0.75$ ) y, por tanto, habría que efectuar un nuevo *Rehashing*, que la solución de este ejercicio NO contempla.**

Parte 2 de 2 - Histograma de ocupación de una Tabla Hash

4.5/ 4.5 Puntos

## Preguntas 6 de 8

1.5/ 1.5 Puntos

Se dispone de una Tabla *Hash* Enlazada de capacidad 10, claves de tipo *Integer* y, por tanto, con una función de dispersión  $\text{indiceHash}(c) = c \% 10$ . Completa el histograma de ocupación de la Tabla tras realizar las siguientes inserciones en una Tabla inicialmente vacía y sin hacer *Rehashing*: 31, 34, 33, 53, 14, 13, 20, 56, 71, 15, 23, 3, 62.

| Longitud de la cubeta             | 0  | 1  | 2  | 3  | 4  | 5  | 6  |
|-----------------------------------|----|----|----|----|----|----|----|
| Número de cubetas de esa longitud | ✓3 | ✓4 | ✓2 | ✓0 | ✓0 | ✓1 | ✓0 |

Respuesta correcta: 3, 4, 2, 0, 0, 1, 0

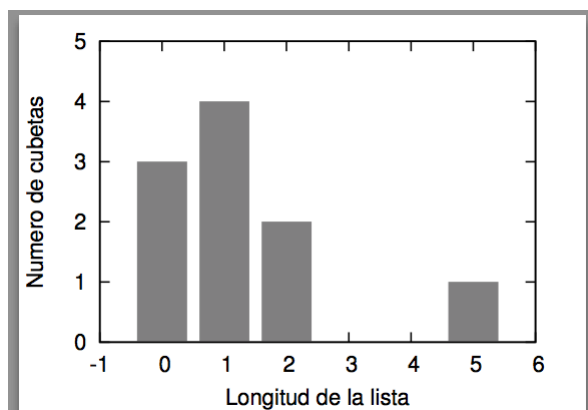
## Comentarios:

La Tabla después de las inserciones tiene  $\text{talla} = 13$  y  $\text{FC} = 1.3$ , pues no se hace *Rehashing*, y su contenido es:

| Índice de cubeta | Lista de claves (cubeta) |
|------------------|--------------------------|
| 0                | 20                       |
| 1                | 31, 71                   |
| 2                | 62                       |
| 3                | 33, 53, 13, 23, 3        |
| 4                | 34, 14                   |
| 5                | 15                       |
| 6                | 56                       |
| 7                |                          |
| 8                |                          |
| 9                |                          |

Por lo tanto, hay 3 cubetas de longitud 0, 4 de longitud 1, 2 de longitud 2 y 1 de longitud 5.

El histograma de ocupación sería entonces:




## Preguntas 7 de 8

1.5/ 1.5 Puntos

¿Es posible realizar inserciones en la Tabla *Hash* de la pregunta anterior para que su histograma de ocupación sea el que aparece en la figura adjunta?

En caso afirmativo, piensa en valores concretos de números enteros tales que, al insertarlos, se obtiene una Tabla *Hash* con dicho histograma. En caso negativo, piensa por qué no es posible insertar más elementos en la Tabla.

 [dat2.pdf](#) 6 KB

☐ Verdadero  
☒ Falso

**Respuesta correcta:**Falso

**Comentarios:**

La diferencia entre este histograma y el anterior es que en él hay una cubeta más de longitud 0 (vacía). Como **no es posible modificar mediante inserciones y sin *Rehashing*** el número de cubetas de una Tabla *Hash* (ni el de las de longitud 0 ni el de las de cualquier otra longitud), resulta imposible que mediante inserciones el histograma de la pregunta anterior pase a ser el que aparece en la figura adjunta.

## Preguntas 8 de 8

1.5/ 1.5 Puntos

¿Es posible realizar borrados en la Tabla *Hash* de la primera pregunta de esta parte para que su histograma de ocupación sea el que aparece en la figura que se adjunta?

En caso afirmativo, piensa en valores concretos de números enteros tales que, al borrarlos de la Tabla, se obtiene una Tabla *Hash* con dicho histograma. En caso negativo, piensa por qué no es posible eliminar elementos de la Tabla.

 [dat3.pdf](#) 6 KB

☐ Verdadero  
☒ Falso

**Respuesta correcta:**Verdadero

**Comentarios:**

En la Tabla/histograma inicial hay: 3 cubetas de longitud 0, 4 de longitud 1, 2 de longitud 2 y 1 de longitud 5.

En la Tabla/histograma adjunto hay: 4 cubetas de longitud 0, 4 de longitud 1, 1 de longitud 2 y 1 de longitud 5.

Esta diferencia puede ocurrir porque una de las cubetas de longitud 2 se ha vaciado, i.e. porque se han borrado el 31 y el 71 o el 34 y el 14.

También puede ser que se haya vaciado alguna de las 4 cubetas de longitud 1 (porque se ha borrado el 20 o el 62, o el 15 o el 56) y, además, se haya eliminado un elemento de alguna de las 2 cubetas de longitud 2 (bien el 31 o el 71, o bien el 34 o el 14).

Por tanto, sí es posible que mediante borrados el histograma inicial pase a ser el que aparece en la figura adjunta.

- PoliformaT
- UPV
- Powered by Sakai
- Copyright 2003-2020 The Sakai Foundation. All rights reserved. Portions of Sakai are copyrighted by other parties as described in the Acknowledgments screen.