## **Evaluación teórica - Ordenamiento**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Realice esta evaluación y logre obtener puntos de ¡**Experiencia**, **Competencia** y **Habilidad**! ¡Adelante!

Este desafío busca evaluar la comprensión e interpretación que usted ha logrado de los contenidos del módulo 3.

Para poder responder estas preguntas, idealmente debe revisar los contenidos teóricos y prácticos presentados en las primeras secciones en donde se explican los conceptos básicos de ordenamiento. La estructura de esta evaluación es de selección múltiple, en la que usted deberá:

* Analizar varias implementaciones de diferentes algoritmos en sintaxis Java simple.
* Responder preguntas de análisis basados en el código presentado.
* Identificar los elementos correctos e incorrectos del código y proponer correcciones a los mismos.
* Comprender la ejecución del algoritmo dado.
* Proponer elementos de mejora de eficiencia sobre el código.

El desafío consta de 12 preguntas. Cada una tiene varias opciones de respuesta pero sólo una es correcta. Al final recibirá la retroalimentación de su desempeño.

Responda la siguiente pregunta basándose en el fragmento de código de un algoritmo de ordenamiento de burbuja para números enteros, el cual se presenta a continuación:

01 for( int i = x; i > 0; i-- )

02 {

03 for( int j = 0; j < i; j++ )

04 {

05 if( arreglo[ j ] > arreglo[ j + 1 ] )

06 {

07 int temp = arreglo[ j ];

08 arreglo[ j ] = arreglo[ j + 1 ];

09 arreglo[ j + 1 ] = temp;

10 }

11 }

12 }

13 return arreglo;

La función o rol del ciclo externo en el código presentado es:

1 punto

Intercambia los valores numéricos de a pares si se cumple que no están en el orden deseado.

Lleva la cuenta de las veces que es necesario iterar como máximo con el ciclo interno para ordenar el arreglo

Intercambia los valores numéricos de a pares si se cumple que están en el orden deseado.

Lleva la cuenta de las veces que es necesario iterar como mínimo con el ciclo interno para ordenar el arreglo.

### 2.

Pregunta 2

Suponga que usted tiene un arreglo desordenado con el siguiente estado inicial:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | 20 | 3 | 7 | 1 | 9 | 8 | 16 | 10 | 21 |

Basándose en el siguiente fragmento de código de un algoritmo de ordenamiento parcial de burbuja para un arreglo de números enteros, imagine cómo completar el ciclo interno para que compare e intercambie los elementos del arreglo (cuando sea necesario) de forma tal que quede ordenado descendentemente.

        int a = 2\* arreglo.length/3;

        for ( int i = 0; i < a; i++ )

        {

            for ( int j = 0; j < a-i-1; j++ )

            {

              //TODO Aquí irían sus nuevas instrucciones

            }

        }

Una vez tenga su propuesta de solución (puede ser en papel), responda a la siguiente pregunta

¿Cuál es el arreglo ordenado resultante?

1 punto

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 21 | 20 | 16 | 15 | 10 | 9 | 8 | 7 | 3 | 1 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 20 | 15 | 9 | 7 | 3 | 1 | 8 | 16 | 10 | 21 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 7 | 9 | 15 | 20 | 8 | 16 | 10 | 21 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 20 | 15 | 7 | 3 | 1 | 9 | 8 | 16 | 10 | 21 |

### 3.

Pregunta 3

Suponga que usted tiene un arreglo desordenado con el siguiente estado inicial:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | 2 | 13 | 7 | 1 | 9 | 8 | 19 | 10 | 21 |

Tomando como base el siguiente fragmento de código de un algoritmo de ordenamiento por inserción ascendente, y teniendo en cuenta que se quiere modificar el algoritmo para que cuando haya un intercambio de elementos en el arreglo sume los dos valores y acumule el resultado en una variable, imagine los cambios necesarios para cumplir con el nuevo requerimiento. Suponga que "arreglo" es un atributo de la clase dueña del método, que contiene los valores mostrados anteriormente.

public int ordenarArregloInsercion()

{

//Aquí debería declararse la variable de tipo entero donde se acumulará el resultado.

//Aquí deberían declararse las variables de tipo entero a, b y c que definen los límites de los dos ciclos. No olvide que es un ordenamiento por inserción.

for ( int i = a; i < b; i++ )

{

boolean termino = false ;

for ( int j = i; j > c && !termino ; j-- )

{

if ( arreglo[j] < arreglo[j-1] )

{

//Piense en las modificaciones necesarias para tener en cuenta el nuevo requerimiento

int temp = arreglo [j-1];

arreglo[j-1] = arreglo[j];

arreglo[j]= temp;

}

else

{

termino = true;

}

}

}

//Aquí se retorna la suma

}

Una vez tenga su propuesta de solución (puede ser en papel), responda la siguiente pregunta:

¿Cuál es la suma resultante?

1 punto

207

105

414

312

### 4.

Pregunta 4

Responda esta pregunta a partir del siguiente fragmento de código que representa un algoritmo de ordenamiento por inserción:

01    for( int i = 1; i < arreglo.length; i++ )

02    {

03        for( int j = i; j > 0 && arreglo[ j - 1 ] > arreglo[ j ]; j-- )

04        {

05            int temp = arreglo[ j ];

06            arreglo[ j ] = arreglo[ j - 1 ];

07            arreglo[ j - 1 ] = temp;

08        }

09    }

¿Cómo se podría expresar en lenguaje natural la iteración interna?

1 punto

Mientras se esté recorriendo el subarreglo ordenado que va de 0 a i y siempre que el elemento en la posición j-1 sea mayor que el elemento en la posición j.

Mientras j sea mayor que 0 y los elementos en la posición j y j-1 no estén ordenados adecuadamente.

Mientras j sea mayor que 0 y el elemento en la posición j-1 sea mayor que el elemento en la posición j.

Mientras se esté recorriendo el arreglo no ordenado entre 0 e i, y siempre que el elemento en la posición j-1 sea mayor que el elemento en la posición j.

### 5.

Pregunta 5

Suponga que usted tiene un arreglo desordenado con el siguiente estado inicial:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | 23 | 15 | 7 | 1 | 9 | 4 | 13 | 17 | 21 |

Después de la tercera iteración del ciclo externo de un algoritmo de ordenamiento, el arreglo queda de la siguiente manera:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 7 | 8 | 23 | 11 | 9 | 15 | 13 | 17 | 21 |

¿Cuál fue el algoritmo usado para ordenar este arreglo?

1 punto

Inserción descendente

Burbuja ascendente

Selección ascendente

Inserción ascendente

### 6.

Pregunta 6

Suponga que usted tiene un arreglo desordenado con el siguiente estado inicial:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 23 | 15 | 7 | 14 | 9 | 1 | 13 | 9 | 21 |

Después de la tercera iteración del ciclo externo de un algoritmo de ordenamiento, el arreglo queda de la siguiente manera:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 23 | 15 | 14 | 13 | 9 | 21 | 9 | 7 | 1 |

¿Cuál fue el algoritmo usado para ordenar este arreglo?

1 punto

Selección ascendente

Ninguna de las anteriores es válida

Inserción descendente

Burbuja descendente

Selección descendente

### 7.

Pregunta 7

Suponga que usted tiene un arreglo desordenado con el siguiente estado inicial:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 23 | 11 | 7 | 8 | 9 | 1 | 13 | 9 | 21 |

Se decide aplicar el algoritmo de inserción ascendente. ¿Cuál es el estado del arreglo al finalizar la cuarta iteración del ciclo externo?

1 punto

Ninguna de las anteriores

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 10 | 8 | 11 | 23 | 9 | 1 | 13 | 9 | 21 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 10 | 11 | 8 | 23 | 9 | 1 | 13 | 9 | 21 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 10 | 11 | 23 | 8 | 9 | 1 | 13 | 9 | 21 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 8 | 10 | 11 | 23 | 9 | 1 | 13 | 9 | 21 |

### 8.

Pregunta 8

Suponga que usted tiene un arreglo desordenado con el siguiente estado inicial:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 23 | 17 | 7 | 18 | 9 | 11 | 13 | 8 | 21 |

Se decide aplicar el algoritmo de burbuja descendente. ¿Cuál es el estado del arreglo al finalizar la segunda iteración del ciclo externo?

1 punto

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 23 | 17 | 10 | 18 | 7 | 9 | 11 | 13 | 8 | 7 |

Ninguna de las anteriores

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 17 | 7 | 18 | 9 | 23 | 11 | 21 | 8 | 7 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 17 | 7 | 18 | 23 | 9 | 11 | 13 | 8 | 7 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 23 | 17 | 18 | 10 | 11 | 13 | 9 | 21 | 8 | 7 |

### 9.

Pregunta 9

Suponga que usted tiene un arreglo desordenado con el siguiente estado inicial:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 23 | 15 | 7 | 14 | 9 | 1 | 13 | 9 | 21 |

Igualmente, cuenta con el siguiente fragmento del código de un programa Java. Suponga que “arreglo” es un atributo de la clase dueña del método, que contiene los valores mostrados anteriormente.

public void ordenarArreglo1()

{

for ( int i = 0; i < arreglo.length; i += 2 )

{

int menor = arreglo[i];

int pos = i;

for ( int j = i+2; j < arreglo.length; j+=2)

{

if ( arreglo[j] < menor )

{

menor = arreglo[j];

pos = j;

}

}

int temp = arreglo[i];

arreglo[i] = menor;

arreglo[pos] = temp;

}

}

public void ordenarArreglo2()

{

for ( int i = 3; i < arreglo.length; i+=2 )

{

boolean termino = false;

for ( int j = i; j > 1 && !termino; j -= 2 )

{

if ( arreglo[j] < arreglo[j-2] )

{

int temp = arreglo [j-2];

arreglo[j-2] = arreglo[j];

arreglo[j] = temp;

}

else

termino = true;

}

}

}

public void metodo3()

{

ordenarArreglo1();

ordenarArreglo2();

}

Con base en el arreglo y el código anterior, responda la siguiente pregunta:

¿Qué tarea está realizando el método llamado “método 3”?

1 punto

Ordenando las posiciones impares del arreglo con el método de selección y las posiciones pares con el método de inserción.

Ordenando las posiciones impares del arreglo con el método de inserción y las posiciones pares con el método de selección.

Ordenando el arreglo usando el método de selección.

Ordenando el arreglo usando el método de inserción.

### 10.

Pregunta 10

¿Cuál de los algoritmos de ordenamiento vistos es el más eficiente en el peor de los casos?, entendiendo por “el peor de los casos” el caso en el que ningún elemento del arreglo en su estado inicial se encuentra en la posición en la que quedaría después del ordenamiento.

1 punto

Ordenamiento por inserción

Ordenamiento por burbuja

Es indiferente, todos tienen la misma complejidad

Ordenamiento por selección

### 11.

Pregunta 11

Se tiene un arreglo de 15 posiciones y se conoce que las primeras 10 posiciones iniciales del arreglo están ordenadas y las demás posiciones están desordenadas. Se desea usar el algoritmo de inserción para ordenar el arreglo ¿Cuáles serían los límites de los ciclos para que el algoritmo haga el número mínimo de iteraciones?

1 punto

Ciclo externo: Desde la posición 0 hasta la última posición del arreglo

Ciclo interno: Desde la posición actual del ciclo externo hasta la posición 0 del arreglo

Ciclo externo: Desde la posición 10 hasta la última posición del arreglo

Ciclo interno: Desde la posición actual del ciclo externo hasta la posición 0 del arreglo.

Ciclo externo: Desde la posición 0 hasta la penúltima posición del arreglo

Ciclo interno: Desde la posición anterior a la posición actual del ciclo externo (actual-1) hasta la posición 0 del arreglo

Ciclo externo: Desde la posición 11 hasta la última posición del arreglo

Ciclo interno: Desde la posición anterior a la posición actual del ciclo externo (actual-1) hasta la posición 0 del arreglo

### 12.

Pregunta 12

Se tiene un arreglo de 5 posiciones y se sabe que el elemento de menor valor está en la última posición del arreglo. Si se utiliza el algoritmo de selección ascendentemente, ¿en cuántas iteraciones del ciclo externo el elemento de menor valor queda en la primera posición del arreglo?

1 punto

25

12

0

1