**INSTITUTO DE EDUCACIÓN SUPERIOR CIBERTEC**

**DIRECCIÓN ACADÉMICA**

**CARRERA PROFESIONALES**

**CURSO :** Algoritmos y Estructura de Datos

### NOTA

**PROFESOR** **:** **Erick Dejo Vicente**

**SEMESTRE :** 2023 – 01

**CICLO :** Segundo

**SECCIÓN** **:** T2IL

**GRUPO** **:** 01

**DURACIÓN :**  50 minutos

**FECHA** **:**

|  |
| --- |
| **ALUMNO (A) : Raúl Ichiro Rosas Chinen** |

**CASO DE LABORATORIO 2**

**Consideraciones generales:**

* La solución debe ser presentada en el archivo **AED\_Solucion\_ApellidosNombres\_CL2.pdf**.
* El color del texto debe ser legible, de lo contrario, la solución no será calificada.
* Lea cuidadosamente la rúbrica para cumplir con todo lo solicitado.
* Si dos o más alumnos presentan la misma solución, recibirán la nota 99 (cero anulado).

**LOGRO**

Al término del curso, el alumno implementa los conceptos de encapsulamiento, herencia y polimorfismo de la programación orientada a objetos en la construcción de aplicaciones java básicas que realizan mantenimiento de datos almacenados en arreglo de objetos y archivos planos.

**Consolidado**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pregunta** | **Puntaje** | | **Llenar solo en caso de recalificación justificada** | |
| **Máximo** | **Obtenido** | **Sustento** | **Puntaje** |
| 1 | 20 |  |  |  |
|  |  |  |  |  |
| **Nota Recalificada** | | | |  |

Problema (20 puntos)

Dada la clase **ArregloPrecios** ubicada en el paquete ***examen***:

**package** examen;

**public** **class** ArregloPrecios {

**private** **double** precio[];

**private** **int** indice;

**public** ArregloPrecios() {

precio = **new** **double**[10];

indice = 0;

}

**public** **int** tamaño() {

**return** indice;

}

**public** **double** obtener(**int** p) {

**return** precio[p];

}

**public** **void** adicionar(**double** s) {

**if** (indice == tamaño())

ampliarArreglo();

precio[indice] = s;

indice++;

}

**private** **void** ampliarPrecio() {

**double** aux[] = precio;

precio = **new** **double**[indice + 10];

**for** (**int** i = 0; i < indice; i++)

precio[i] = aux[i];

}

**public** **void** eliminarTodo() {

indice = 0;

}

**public** **void** eliminarFinal() {

indice--;

}

}

Considerando que el arreglo será ingresado progresivamente con datos tomados de la GUI y que podría no encontrarse completamente lleno, implemente:

En la clase **ArregloPrecios**:

* Un método denominado **posicionUltimoPrecioMenorQue900**, que busque y retorne la posición del último precio menor que 900. En caso que no exista, retornar -1.
* Un método denominado **sumaPreciosMayoresQue750**, que retorne la suma de todos los precios mayores que 750.
* Un método denominado **reemplazarUltimoPrecioMenorQue900**, que reemplace el último precio menor que 900, por la suma de los precios mayores que 750. Si el reemplazo es posible, retorne **true**; en caso contrario, retorne **false**.
* Un método denominado **incrementarPreciosMenoresQue1100** que incremente en 14% todos los precios menores que 1100 y retorne la cantidad de incrementos efectuados.
* Un método denominado **eliminarUltimoPrecioMenorQue900**, que elimine el último precio menor que 900. Si la eliminación es posible, retorne **true**; en caso contrario, retorne **false**.

En la clase **Problema**:

* Ante la pulsación del botón **Reemplazar último precio menor que 900**, invoque al método **reemplazarUltimoPrecioMenorQue900** e invoque al método **listar**. En caso de que no exista ningún precio menor que 900, muestre el mensaje "No existe ningún precio menor que 900".
* Ante la pulsación del botón **Incrementar precios menores que 1100** invoque al método **incrementarPreciosMenoresQue1100** e invoque al método **listar**. Muestre, además, un mensaje indicando cuantos precios fueron incrementados.
* Ante la pulsación del botón **Eliminar último precio menor que 900**, invoque al método **eliminarUltimoPrecioMenorQue900** e invoque al método **listar**. En caso de que no exista ningún precio menor que 900, muestre el mensaje "No existe ningún precio menor que 900".

**RÚBRICA**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problema** | **20.0 puntos** | **15.0 puntos** | **10.0 puntos** | **5.0 puntos** |
| 20  Puntos | Implementa correctamente los ocho métodos solicitados. | Implementa correctamente seis métodos solicitados. | Implementa correctamente cuatro métodos solicitados. | Implementa correctamente dos métodos solicitados. |