Práctica 3
Polinomio de direccionamiento

Estructuras de datos

1. Meta

Que el alumno domine el manejo de información almacenada en arreglos multidimensionales.

2. Objetivos

Al finalizar la práctica el alumno será capaz de:

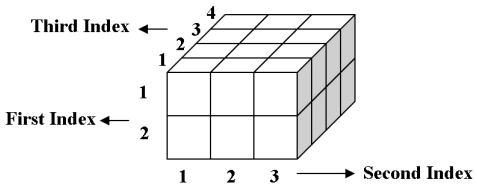
- Almacenar un arreglo de n dimensiones en uno de una sola dimensión.

3. Antecedentes

Los arreglos multidimensionales son aquellos que tienen más de una dimensión, los más comunes son de dos dimensiones conocidos como *matrices*. Este tipo de arreglos en Java se ven como arreglos de arreglos.

Existen dos momentos fundamentales en la creación de arreglos:

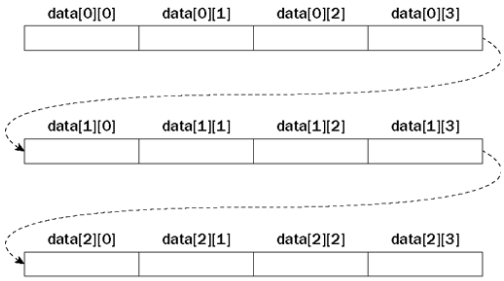
1. Declaración: en esta parte no se reserva memoria, solo se hace una referencia.
```
int [][] arreglo;
float[][][] b;
```
2. Reservación de la memoria y la especificación del número de filas y columnas.
```
arreglo = new int[10][5];//matriz con 10 filas y 5 columnas
b = new float [13][25][4];//cubo con 13 filas, 25 columnas y 4 planos
```



Three-dimensional array with twenty four elements

Nota: Se puede declarar una dimensión primero, pero siempre debe de ser en orden, por ejemplo `arreglo = new int[][10];` es erróneo pues las filas quedan indeterminadas.

Dado que la memoria de la computadora es esencialmente lineal es natural pensar en almacenar los elementos de un arreglo multidmensional en un arreglo de unidemsional. Por ejemplo, consideren un arreglo de tres dimensiones:



The array elements are stored in contiguous locations in memory.

Generalicemos el ejemplo anterior en uno de n-dimensiones, donde el tamaño de cada dimensión i esta dada por δ_i, entonces tenemos un arreglo n-D: $\delta_1 \times \delta_2 \times ... \times \delta_n$.

Entonces la posición del elemento a$[i_1][i_2][...][i_n]$ esta dada por:

$$\sum_{i=1}^{n} f_j i_j$$

donde $f_j = \begin{cases} 1 & \text{si } j{=}n \\ \prod_{k=j+1}^{n} \delta_k & \text{si } 1 \leq j < n \end{cases}$

4. DESARROLLO

La práctica consiste en implementar los métodos definidos en la interfaz **Arreglo**, la cual convierte un arreglo de enteros de n-dimensiones en uno de una dimensión. El constructor de la clase que implemente la interfaz deberá tener como parámetro un arreglo de ints que representen las dimensiones del arreglo. Por ejemplo para crear un arreglo tridimensional ($3 \times 10 \times 5$), invocamos el constructor de la siguiente forma:

```
1  Arreglo a = new Arreglo(new int [] {3,10,5};
```

```
1  /*
2   * Codigo utilizado para el curso de Estructuras de Datos.
3   * Se permite consultarlo para fines didacticos en forma personal,
4   * pero no esta permitido transferirlo tal cual a estudiantes actuales o potenciales.
5   */
6  package practica3;
7
8  public interface ArregloInterface{
9
10         /**
11         * Devuelve el elemento que se encuentra en la posicion <code>th</code>
12         * en el arreglo multidimensional.
13         * @param indices arreglo con los indices del elemento a recuperar.
14         * @return el elemento almacenado en la posicion <code>i</code>.
15         */
16         public int obtenerElemento(int [] indices);
```

```
17
18          /**
19          * Asigna un elemento en la posicion <code>th</code> del arreglo multidimensional.
20          * @param indices arreglo con los indices donde se almacenara el elemento.
21          * @param elem elemento a almacenar.
22          */
23          public void almacenarElemento(int [] indices, int elem);
24
25          /**
26          * Devuelve la posicion <code>i</code> del elemento en el arreglo de una dimension.
27          * @param indices arreglo con los indices donde esta el elemento en el arreglo
28          * multidimensional.
29          * @return la posicion del elemento en el arreglo de una dimension.
30          * @throws IndexOutOfBoundsException si alguno de los indices del arreglo no esta
31          * dentro del rango.
32          */
33          public int obtenerIndice(int [] indices);
34
35  }
```

5. Preguntas

1. Explica la estructura de tu código, explica en más detalle tu implementación del método obtenerIndice.
2. ¿Cuál es el orden de complejidad de cada método?

6. Forma de entrega

- Asegúrense de borrar todos los archivos generados, incluyendo archivos de respaldo usando `ant clean`.
- Copien el directorio `Practica3` dentro de un directorio `<apellido1_apellido2>` donde `apellido1` es el primer apellido de un miembro del equipo y `apellido2` es el primer apellido del otro miembro. Por ejemplo: `marquez_vazquez`.
- Borren el directorio libs en la copia.
- Agreguen un archivo `reporte.pdf` dentro del directorio `Practica3` de la copia, con el nombre completo de los integrantes del equipo, las repuestas a las preguntas de la sección anterior y cualquier comentario que quieran hacer sobre la práctica.
- Compriman el directorio `<apellido1_apellido2>` en un archivo `<apellido1_apellido2>.tar.gz`. Por ejemplo: `marquez_vazquez.tar.gz`.
- Suban este archivo en la sección correspondiente del aula virtual. Basta una entrega por equipo.

7. Forma de evaluación

Para su calificación final se tomarán en cuenta los aspectos siguientes:

60 % Calificación generada por las pruebas automáticas. Usaremos nuestros archivos, por lo que si realizan modificaciones a sus copias, éstas no tendrán efecto al momento de calificar.

25 % Documentación completa y adecuada. Entrega en el formato requerido, sin archivos `.class`, respaldos, bibliotecas de `JUnit` u otros no requeridos.

15 % Revisión manual del código, para verificar que se cumpla con las especificaciones, que no se haya copiado, etcétera.