PROGRAMACIÓN

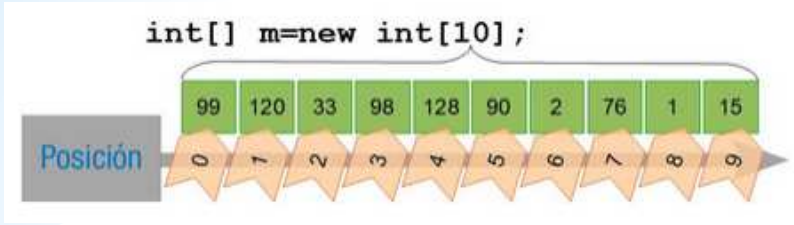
SEGUNDO TRIMESTRE

Tutoría colectiva (2)

ESTRUCTURAS DE ALMACENAMIENTO

<table>
<tr><td>Simples</td><td colspan="2">Datos primitivos:
int, char, float, double, boolean, ..</td></tr>
<tr><td rowspan="3">Complejos</td><td colspan="2">Clases</td></tr>
<tr><td rowspan="2">Estructuras de almacenamiento</td><td>Estáticas: arrays</td></tr>
<tr><td>Dinámicas: colecciones</td></tr>
</table>

ARRAYS



* Permiten albergar datos del **mismo tipo**.

* Su longitud se establece durante su creación.

* Una vez establecida su longitud, ya no se puede modificar.

* Un elemento de un array es el valor de una de sus posiciones y se puede acceder mediante un índice.

* En Java es un tipo de clase especial que hereda implícitamente de la clase java.lang.Object

ARRAYS.- Declaración y Creación

* **Declaración**: Se realiza indicando el tipo de datos que va a contener y los corchetes. Pueden contener tanto tipos primitivos (int, double, etc..) como tipos completos (clases).

```
modificadorAcceso tipoDato[]  nombreArray [=valorInicial];
    private int[]   numeros;          //Array de enteros
    private String[]  cadenas;        //Array de Strings
    protected Alumno[]   alumnos;             //Array de objetos Alumno
```

* **Creación**:

```
nombreArray = new tipoDato[longitud];
    numeros = new int[5];
    cadenas = new String[10];
    alumnos = new Alumno[30] ;
    private int[] numeros = new int[5];
```

Al igual que con otros objetos, podemos declararlos y crearlos en una sola línea.

ARRAYS.- Inicialización y Acceso

* Una vez creado el array, todas sus posiciones son inicializadas al valor por defecto (0, 0.0, false o null si es de tipo complejo).

* Se pueden inicializar todos los elementos de un array con un valor determinado, igualándolo a una lista de valores separados por comas y entre llaves { }

```
private int[] numeros = {1,2,3,4,5};          //array de 5 enteros
private String[] cadenas = {"hola","adiós"}; //array de 2 Strings
```

* El tipo de índice para acceder a un array es *int*.
* La primera posición es la 0.

```
numeros[0] = 1;
cadenas[1] = "adiós";
alumnos[0] = new Alumno('Juan García');
```

* Podemos conocer el número de elementos de un array, accediendo a su atributo *length*.

```
alumnos.length
```

* No podemos eliminar ni insertar posiciones. Si intentamos acceder a una posición fuera del array se lanzará una excepción (*ArrayIndexOutOfBoundException*)

ARRAYS.- Ejemplo

```
int[] edadTrabajador = new int[100];
Scanner teclado= Scanner (System.in);
for ( int  i = 0 ; i  < 100 ; i++){
      System.out.println(“Introduce la edad para el trabajador”);
       edadTrabajador [ i ] = teclado.nextInt();
}
//…
int[] alturaTrabajador = new int[100];
for ( int  i = 0 ; i  < 100 ; i++){
      System.out.println(“Introduce la altura para el trabajador”);
      alturaTrabajador[ i ] = teclado.nextInt();
}
```

ARRAYS.- Bucle for each

* En las últimas versiones, Java incorpora una sentencia que facilita la iteración para los elementos de cualquier tipo de colección (arrays, listas, etc..)

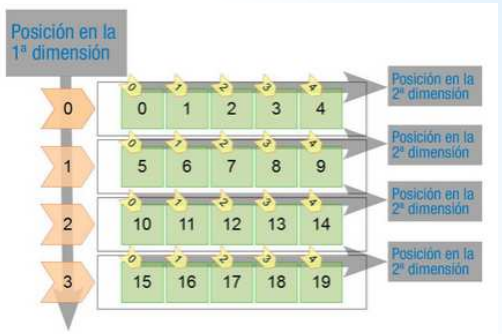
```
for(inicialización: colección){
      sentencias;
}
```

```
public void listar(int[] tabla){
    for(int elemento: tabla){
        System.out.println(elemento);
    }
}
```

La variable del bucle será del mismo tipo que el array.

```
1  import java.util.Scanner;
2  public class pruebaArray {
3    public static void main (String[] args) {
4        String[] cadenas =new String[4];
5        Scanner teclado = new Scanner(System.in);
6        //en un bucle for leemos una cadena en cada pasada
7        for(int i=0; i<cadenas.length; i++){
8            System.out.print("Introduce la cadena["+i+"]= ");
9            cadenas[i]=teclado.nextLine();
10       }
11       //recorremos el array y mostramos sus valores
12       System.out.println("Las cadenas introducidas son: ");
13       for (String cadena: cadenas)
14           System.out.println(cadena);
15   }
16 }
```

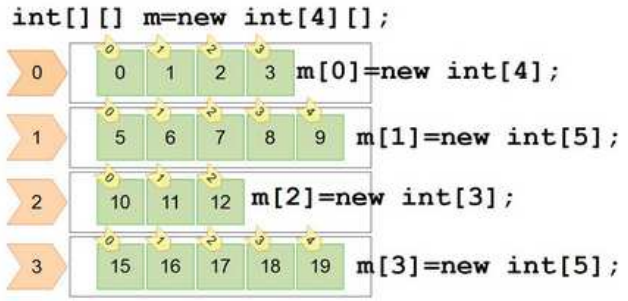
ARRAYS MULTIDIMENSIONALES



* **Regulares**: un array que contiene arrays todos del mismo tamaño.

modificadorAcceso tipoDato[][] nombreArray = new tipoDato[long][long];

```
private int[][]   numeros= new int[4][5]      //4 filas y 5 columnas
```



* **Irregulares**: un array que contiene arrays de distintos tamaños.

* Se pueden inicializar con listas de valores, al igual que los unidimensionales:

```
private int[][]  numeros = {{1,2,3},{4,5,6}};
```

ARRAYS MULTIDIMENSIONALES.- Ejemplo

```
3   public static void mVisualizar(int [][] m)    {
4       for(int f=0;f<m.length;f++) {
5           for(int c=0;c<m[f].length;c++)
6               System.out.print(m[f][c]+" ");
7           System.out.println();
8       }
9
10  }
11  public static void mSumarFilas(int [][] m) {
12      int sumaf=0;
13      for(int f=0;f<m.length;f++)  {
14          for(int c=0;c<m[f].length;c++)
15              sumaf+=m[f][c];
16          System.out.println("la suma de la fila: "+f+"  es "+sumaf);
17          sumaf=0;
18      }
19  }
20  public static void main(String[] args) {
21      int [][] A;
22      A=new int [3][2];
23
24      mCargar(A);  //no está implementado en el ejemplo
25      mVisualizar(A);
26      mSumarFilas(A);
27  }
28
```

Utilizamos dos bucles `for` anidados. Uno para recorrer las filas (índice `f`) y otro para recorrer las columnas (índice `c`)

`m.length` devuelve el número de filas

`m[f].length` devuelve el número de elementos de la fila `f`.

OPERACIONES CON ARRAYS

* Búsqueda en arrays desordenados:

```
i=0;
while( i<TAM-1 && vector[i] != x){
        i++;
  }//fin while

//Imprimo el resultado.
  if ( vector[i]== x )
        System.out.println ("Elemento encontrado en la posicion de indice " + i );
  else
        System.out.println ("Elemento no encontrado.");
```

* Búsqueda en arrays ordenados:
  * Secuencial.
  * Dicotómica o binaria.

```
i=0;
while ( vector[i] < x ){
        i++;
}//fin while
```

OPERACIONES CON ARRAYS

* Búsqueda en arrays ordenados:
  * Dicotómica o binaria.

```
int izq=0;
int der=TAM-1;
cen=((izq+der)/2);          // se calcula el centro del vector
while(vector[cen]!= x  && izq<der)  {
        if(vector[cen]< x )
              izq=cen+1;        // se cambia el límite izquierdo
        else
              der=cen-1;       // se cambia el límite derecho
        cen=((izq+der)/2);    // nuevo centro
}//fin while
//Imprimo el resultado.
if(vector[cen]== x )
        System.out.println ("Elemento encontrado en la posición " + cen+1);
else
        System.out.println ("Elemento no encontrado.");
```

OPERACIONES CON ARRAYS

* Ordenación de arrays: existen muchos métodos para ordenar los elementos de un array. Algunos de ellos son:
  * Ordenación por inserción directa o método de la baraja.
  * Ordenación por selección directa.
  * Ordenación por intercambio directo, método de la burbuja.

```
public static void ordenarBurbuja (int[ ] vector) {
   int i,j,aux;
   for (i=0;i<DIM-1;i++) {
      for (j=0;j<DIM-1;j++)
         if (vector[ j ] > vector[ j+1 ]) {
            aux=vector[j];
            vector[j]=vector[j+1];
            vector[j+1]=aux;
         }
   }
}
```