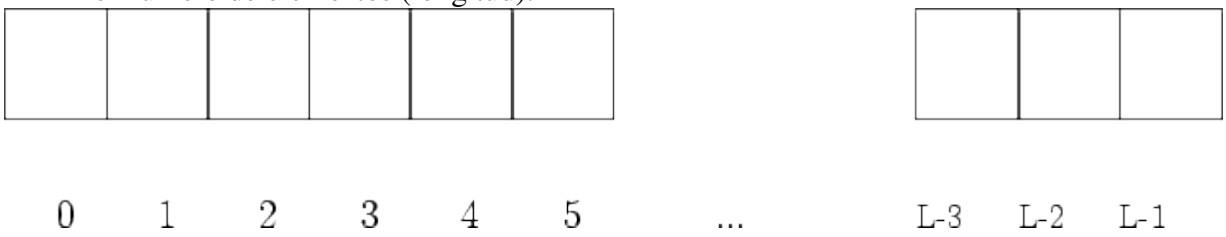
Unidad Didactica 4:
Los Arrays

- Un ``Array'' es una lista de elementos del mismo tipo (homogénea), cada uno ellos identificado por un índice.
- Se puede acceder a los elementos del Array (para guardar o recuperar la información) en cualquier orden.
- El Array queda definido por
  - el tipo básico de elementos que aglutina,
  - el número de elementos (longitud).



- Es la estructura de datos que se usa más frecuentemente en programación.
- La gestión de información con el Array es eficiente. De hecho, la memoria del ordenador se gestiona como un Array en los lenguajes de bajo nivel.
- Los Arrays se utilizan con tanta frecuencia que todos los lenguajes de programación ofrecen algún modo para utilizarlos con facilidad.

Arrays en Java

- En Java, los Array pueden ser de cualquier tipo de dato, incluidos objetos.
- El tipo de dato Array es, a su vez, un objeto.
- Las variables del tipo Array se declaran utilizando **[]**, del siguiente modo:

```
tipo_basico[]  nombre_variable
```

- Por ejemplo,

```
int[]  fila;
```

  declara la variable `fila` del tipo Array de datos del tipo `int`.
- Estas variables almacenarán la referencia al objeto.
- Para crear el objeto, se utiliza el operador **new** de la forma:

```
new tipo_basico[numero_de_elementos]
```

- Por ejemplo,

```
fila = new int[100];
```

crea el objeto que puede almacenar 100 enteros.

- Se puede declarar la variable y crear el objeto en la misma instrucción:

```
int[] fila = new int[100];
```

- A cada uno de los elementos del Array se accede con el nombre seguido del índice entre corchetes. El índice varía entre 0 y $L-1$, siendo L el número de elementos que puede almacenar. Así,

```
fila[99] = 1763;
```

almacena el valor `1763` en la última posición del Array `fila`.

- Cada uno de los componentes se puede utilizar como una variable cualquiera del tipo de dato básico que almacena.
- Por ejemplo, el Array `fila` ofrece la 100 variables del tipo `int`:

```
fila[0], fila[1], ... ,fila[98], fila[99]
```

- El objeto Array incluye una variable que es la longitud y que tiene el nombre `length`. La variable es pública y se puede acceder a ella directamente como un campo del objeto.
- En el ejemplo de `fila`, este programa imprime 100:

```
public class p2t2p1{
public static void main(String[] args)
    { int[] fila = new int[100];
    int cuantos = fila.length;
    System.out.println(cuantos);}}
```

Objetos Array

- La variable `length` es del tipo `final`, se asigna en la creación del objeto y no se puede modificar posteriormente.
- Cuando se crea el objeto, todos los elementos toman un valor inicial. Si se trata de tipos básicos toman el valor cero, si son números o caracteres, o el valor `false` si son del tipo `boolean`. Si se trata de referencias a objetos, el valor por defecto es `null`.

- Los Arrays se pueden crear e inicializar al mismo tiempo. Para ello, se utilizan los valores entre llaves con el formato

  ```
  tipo_dato[] variable = { v1 , v2 , v3 , . . . , vN};
  ```

  donde $v1, v2, ..., vN$ son los N valores que se asignan a las posiciones $0, 1, ..., N-1$ del Array.
- Por ejemplo, con la instrucción:

  ```
  double[] primos20 = {2.,3. 5.,7.,11., 13.,17.,19.};
  ```

  se crea el objeto `primos20` como un Array de 8 datos del tipo `double`. En la creación del objeto, `primos20.length` se inicializa con el valor 8, y ya no puede modificarse. El resto de los valores si que se pueden modificar, ya que `primos20[0]`,...,`primos20[7]` son 8 variables del tipo `double`.
- La salida del programa:

  ```
  public class p2t2p4{
  public static void main(String[] args){
   double[] primos20 = {2.,3.,5.,7.,11.,13.,17.,19.};
   for(int i = 0;i<primos20.length;i++) {primos20[i]-=1;
        System.out.print(primos20[i] +" "); }    }}
  ```

  será: `1.0 2.0 4.0 6.0 10.0 12.0 16.0 18.0`

Arrays bidimensionales

- También se puede organizar la información en tablas de dos dimensiones, con dos índices para acceder a los elementos.
- Con la instrucción:

  ```
  int[][]  mat = new int[10][10]
  ```

  se crea una matriz capaz de almacenar 100 elementos del tipo `int` y a ellos se accede con las

variables `mat[i][j]`.

- Se pueden crear tablas de cualquier dimensión, pero hay que llevar cuidado en su uso, ya que la memoria que utilizan crece muy rápidamente. Por ejemplo, la matriz

```
int[][][]  mat3 = new int[100][100][100]
```

utilizará más de 4 Mbytes de memoria para almacenar el millón de enteros.

- Un Array bidimensional es la estructura de datos adecuada para almacenar la información de la matrices que se utilizan en álgebra. Los Arrays tridimensionales, son los apropiados para almacenar información de tensores.

- Ilustramos su uso con el programa siguiente

```
public class p2t2p6{
    public static void main(String[] args){
        int[][]  mat = new int[3][5];
        for(int i=0;i<mat.length;i++)
            for (int j=0;j<mat[i].length;j++)
                mat[i][j] = (i+1)*10 + (j+1);
         for(int i= 0;i<mat.length;i++)  {
              for (int j=0;j<mat[i].length;j++)
                System.out.print(mat[i][j]+ "  ");
              System.out.println();}     }}
```

que produce la salida:

```
11  12  13  14  15
21  22  23  24  25
31  32  33  34  35
```

- El número de filas está almacenado en `mat.length` y el número de columnas de la fila k está almacenada en `mat[k].length`