UNIVERSIDAD DE MATANZAS

ARREGLOS UNIDIMENSIONALES

Octubre / 2019

INTRODUCCIÓN A LA PROGRAMACIÓN
INGENIERÍA EN INFORMÁTICA

Objetivos

Caracterizar los arreglos unidimensionales como parte de la construcción de un programa computacional.

Objetivos

Declarar arreglos, inicializarlos y hacer referencia a elementos individuales de los arreglos.

Objetivos

Utilizar la instrucción **for** mejorada para iterar a través de los arreglos.

Pasar arreglos a los métodos como argumentos.

Sumario

- Arreglos
  - Declaración y creación
  - Uso de un inicializador
  - Instrucción **for** mejorada
  - Paso de arreglos a los métodos
- Métodos de la clase *Arrays*

Bibliografía

- *Como programar en Java.*
- *Aprenda Java como si estuviera en primero.*

Arreglos

Los **arreglos** son estructuras de datos que consisten de elementos de datos relacionados, del mismo tipo. Los arreglos son entidades de longitud fija; conservan la misma longitud una vez creados.

Arreglos

Aunque puede reasignarse una variable tipo arreglo de tal forma que haga referencia a un nuevo arreglo de distinta longitud.

Arreglos

En Java, un arreglo es un grupo de variables (llamadas elementos o componentes) que contienen valores, todos del mismo tipo.

Arreglos

Los tipos en Java se dividen en dos categorías: tipos primitivos y tipos de referencia. Los arreglos son objetos, por lo que se consideran como tipos de referencia.

Arreglos

Los elementos de un arreglo pueden ser tipos primitivos o de referencia.
Para hacer referencia a un elemento específico en un arreglo, debemos especificar el nombre de la referencia al arreglo y el número de la posición del elemento en el arreglo.

Arreglos

El número de la posición del elemento se conoce formalmente como el **índice** o **subíndice del elemento**.

Arreglos

Un programa puede hacer referencia a cualquiera de estos elementos mediante **una expresión de acceso a un arreglo** que incluye el nombre del arreglo, seguido por el índice del elemento específico encerrado entre **corchetes** ([ ]).

Arreglos

Declaración y creación

Los objetos arreglo ocupan espacio en memoria. Al igual que los demás objetos, los arreglos se crean con la palabra clave **new** .

Arreglos

Declaración y creación

Para crear un objeto arreglo, el programador especifica el tipo de cada elemento y el número de elementos que se requieren para el arreglo, como parte de una expresión para crear un arreglo que utiliza la palabra clave **new**.

Arreglos

Declaración y creación

Sintaxis

```
<TD> <nombre> []= new <TD> [<capacidad>];
```

<TD>: Tipo de dato
<nombre>: Nombre del arreglo
<capacidad>: Cantidad de elementos del arreglo

Arreglos

Declaración y creación

La siguiente declaración y expresión crea un arreglo, que contiene 12 elementos **int**, y almacena la referencia del arreglo en la variable c

Código

```
int c[] = new int[12];
```

Arreglos

Declaración y creación

También puede realizarse en dos pasos, como se muestra a continuación:

```
int c[ ]; // declara la variable arreglo
c = new int[ 12 ]; /* crea el arreglo; lo asigna a
   la variable tipo arreglo*/
```

Arreglos

Uso de un inicializador

Un programa puede crear un arreglo e inicializar sus elementos con un inicializador de arreglo, que es una lista de expresiones separadas por comas encerrada entre llaves ({ y });

Arreglos

Uso de un inicializador

```
int n[] = { 10, 20, 30, 40, 50 };
```

Crea un arreglo de cinco elementos con los valores de índices 0, 1, 2, 3 y 4 . El elemento n[0] se inicializa con 10, n[1] se inicializa con 20, y así en lo sucesivo.

Arreglos

Instrucción **for** mejorada

Utilizar las instrucciones **for** controladas por un contador para iterar a través de los elementos en un arreglo es muy común.

Arreglos

Instrucción **for** mejorada

La instrucción **for** mejorada, la cual itera a través de los elementos de un arreglo o colección sin utilizar un contador con lo cual, evita la posibilidad de "salirse" del arreglo.

Arreglos

Instrucción **for** mejorada

Sintaxis

```
for ( <parametro> : <nombreArreglo> ){
    <instrucciones>
}
```

<parámetro>: tiene dos partes; un tipo y un identificador (por ejemplo, int numero)
<nombreArreglo>: es el arreglo a través del cual se iterará.

Arreglos

Instrucción **for** mejorada

Sintaxis

```
for ( <parametro> : <nombreArreglo> ){
<instrucciones>
}
```

<parámetro>: tiene dos partes; un tipo y un identificador (por ejemplo, int numero)
<nombreArreglo>: es el arreglo a través del cual se iterará.

Arreglos

Instrucción **for** mejorada

El tipo del parámetro debe concordar con el tipo de los elementos en el arreglo.

Arreglos

Instrucción **for** mejorada

```
public class PruebaForMejorado{
   public static void main( String args[] ){
      int arreglo[] =
         {87,68,94,100,83,78,85,91,76,87};
      int total = 0;
      // suma el valor de cada elemento al total
      for( int numero : arreglo )
         total += numero;
      System.out.printf( "Total de elementos del
         arreglo: %d\n", total );
   }
}
```

Arreglos

Paso de arreglos a los métodos

Para pasar un argumento tipo arreglo a un método, se especifica el nombre del arreglo sin corchetes. Por ejemplo, si el arreglo temperaturasPorHora se declara como

```
double temperaturasPorHora[ ] = new double[ 24 ];
```

entonces la llamada al método

```
modificarArreglo( temperaturasPorHora );
```

Arreglos

Paso de arreglos a los métodos

Todo objeto arreglo “conoce” su propia longitud (a través de su campo *length*). Por ende, cuando pasamos a un método la referencia a un objeto arreglo, no necesitamos pasar la longitud del arreglo como un argumento adicional.

Arreglos

Paso de arreglos a los métodos

Para que un método reciba una referencia a un arreglo a través de una llamada a un método, la lista de parámetros del método debe especificar un parámetro tipo arreglo.

Arreglos

Paso de arreglos a los métodos

Por ejemplo, el encabezado para el método modificarArreglo podría escribirse así:

```
public static void modificarArreglo(int [] b){
//....
}
```

lo cual indica que modificarArreglo recibe la referencia de un arreglo de enteros en el parámetro b.

Arreglos

Paso de arreglos a los métodos

La llamada a este método pasa la referencia al arreglo *temperaturaPorHoras* , de manera que cuando el método llamado utiliza la variable b tipo arreglo, hace referencia al mismo objeto arreglo como temperaturasPorHora en el método que hizo la llamada.

Clase Arrays

Es una clase de utilidad introducida en el JDK 1.2 que contiene métodos static para ordenar, llenar, realizar búsquedas y comparar los arrays clásicos del lenguaje.

Clase Arrays

Para una mejor compresión puedes consultar el libro Cómo programar en Java a partir de la página 794 presente en la bibliografía de la asignatura.

Conclusiones

- Los arrays se crean con el operador **new** seguido del tipo y número de elementos.
- Se puede acceder al número de elementos de un array con la variable miembro implícita *length* (por ejemplo, vect.length).

Conclusiones

- Se accede a los elementos de un array con los corchetes [ ] y un índice que varía de 0 a length-1.
- Se pueden crear arrays de objetos de cualquier tipo. En principio un array de objetos es un array de referencias que hay que completar llamando al operador **new**.

Conclusiones

- Los elementos de un array se inicializan al valor por defecto del tipo correspondiente (cero para valores numéricos, el carácter nulo para char, false para boolean, null para Strings y para referencias).

Conclusiones

- Como todos los objetos, los arrays se pasan como argumentos a los métodos por referencia.
- Se pueden crear arrays anónimos (por ejemplo, crear un nuevo array como argumento actual en la llamada a un método).

UNIVERSIDAD
DE MATANZAS

cosechando el saber

FIN