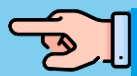
Arreglos y archivos

Introducción a arrays

{desafío} latam_

Utilizar la sintaxis básica del lenguaje Java para la construcción de programas que resuelven un problema de baja complejidad

- Unidad 1: Flujo, ciclos y métodos
- Unidad 2: Arreglos y archivos
- Unidad 3: Programación orientada a objetos
- Unidad 4: Pruebas unitarias y TDD



Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Reconocer los arrays en el lenguaje de programación para sus distintas implementaciones y uso.*
- *Construir programas utilizando arrays para manejar y tratar la información en volúmenes de datos.*

{desafío} latam_

¿Qué entendemos por arreglos?

/* Arrays */

¿Para qué sirven?

- Los arrays se utilizan mucho dentro de la programación.
- Java trata un arreglo como una variable normal, es decir, se:
  - declara
  - inicializa
  - utiliza
- Nos permite resolver diversos tipos de problemas.

Los corchetes [ ]

Indican al compilador que esa variable es un arreglo de un tipo de dato en específico.

Ejemplos:

```
int [] a = {2,4,5,6};
```

```
String[ ] nombre = {"Juan", "Pedro"}; //Array de 2 elementos
```

Posibles usos

- Traer datos de la base de datos en un array y mostrar los datos en una página web.
- Obtener datos desde una API que pueden venir como una colección.
- Traer información de uno o más archivos.

Tipos de arreglos

De tipo de datos primitivos
(int, char ,double, boolean...)

- De tamaño estático, al cual se le define el tamaño al inicializarlo
- Almacenan tipos de datos primitivos como objetos.
- Se almacena en la memoria stack.

De tipo objeto
(Arrays: Arraylist, LinkedList, HashSet, List, entre otros)

- Variables que son trabajadas como objeto y no como variable de dato primitivo.
- Se almacena en la memoria HEAP.
- En tiempo de ejecución, pueden cambiar su tamaño (agregando o quitando elementos).

/* Creando un array */

Estáticos

Debemos especificar el tipo de dato, los corchetes y el nombre de la variable.

Definir el array

Los corchetes pueden estar antes o después del nombre de la variable.

```
int[] a;
int b[];
```

Definir el tamaño del array

Debemos decirle de qué tamaño será dentro de los corchetes.

```
a = new int [4]; //array llamado a, de tipo enteros, de tamaño 4
```

Estáticos

Debemos especificar el tipo de dato, los corchetes y el nombre de la variable.

Introducir un valor

Podemos escribir:

```
a[0] = 4; // en la posición 0 agrega el valor 4
```

Asignar valores iniciales

Al momento de declarar un arreglo:

```
int[] ba = {2,4,5,6};//arreglo llamado ba, de tipo enteros, tamaño 4
```

Dinámicos

Usar ArrayList, agregando previamente la librería **`import java.util.ArrayList;`**

```
ArrayList <Integer> arrayInt = new ArrayList<Integer> ();
```

Índices

- Posición determinada de cada elemento del arreglo.
- Nos permite acceder al elemento que está dentro del arreglo.

Por ejemplo, tenemos:

```
System.out.printf("%d\n", a[0]) // 1
```

Si queremos acceder al primer elemento, debemos acceder a la posición 0.

```
System.out.printf("%d\n", a[0]) // 1
```

Recorrido

- Se debe ocupar sentencias de bucle (for, for each, While).
- Se recorre desde 0 hasta n-1 , donde n es el tamaño del arreglo.
- Con la propiedad length de un arreglo de corchetes podemos obtener el tamaño.

```
int i;
int[] a = {1,2,3,4,5};
int n = a.length;
for(i=0;i<n;i++){
    System.out.printf("%d\n",a[i]);
}
```

Salida:

```
1
2
3
4
5
```

/* Tipos de errores con los índices en un array */

Errores más comunes

Índices

- Se intenta agregar más elementos a un arrays definido y se accede a un índice que no existe; como por ejemplo un índice negativo:

```
int[] a = {1,2,3,4,5}; System.out.printf("%d\n", a[5]);
```

Pero al ejecutar el código obtenemos el error que nos dice que el índice 5 está fuera del límite del arreglo.

```
------------------------------------------------------------------------------
java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 5 at .(#33:1)
```

Errores más comunes

Índices

¿Y si colocamos índices negativos?

```
System.out.printf("%d\n", a[-1]);
```

Ocurre el mismo error:

```
--------------------------------------------------------------------------------
java.lang.ArrayIndexOutOfBoundsException: Index -1 out of bounds for length 5 at .(#34:1)
```

Ejercicio guiado

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

PASO 1:
Creamos un método llamado Suma que retorna un número entero de la suma.

```
// Paso 1
public static int suma() {
}
```

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

PASO 2:
Dentro del método suma, declaramos una variable local llamada suma que parte en cero.

```
public static int suma() {
    //Paso 2
    int suma = 0;
}
```

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

PASO 3:
Inicializamos el arreglo con valores aleatorios.

```
public static int suma() {
     int suma = 0;
     //Paso 3
     int[] arreglo = { 1, 5, 11, 33, 4, 6, 7, 44, 6, 1, -1 };
}
```

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

PASO 4:
Recorremos el arreglo con un ciclo for.

```
public static int suma() {
     int suma = 0;
     int[] arreglo = { 1, 5, 11, 33, 4, 6, 7, 44, 6, 1, -1 };
     //Paso 4
     for (int x = 0; x < arreglo.length; x++) {
     }
}
```

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

PASO 5:

Dentro del ciclo, realizamos condición if donde preguntamos por los valores que están en el intervalo solicitado.

```
public static int suma() {
    int suma = 0;
    int[] arreglo = { 1, 5, 11, 33, 4, 6, 7, 44, 6, 1, -1 };
    for (int x = 0; x < arreglo.length; x++) {
        //Paso 5
        if (arreglo[x] >= 1 && arreglo[x] <= 10) {
        }
    }
}
```

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

PASO 6:

Si la condición se cumple, sumará todos los valores.

```
public static int suma() {
     int suma = 0;
     int[] arreglo = { 1, 5, 11, 33, 4, 6, 7, 44, 6, 1, -1 };
     for (int x = 0; x < arreglo.length; x++) {
          if (arreglo[x] >= 1 && arreglo[x] <= 10) {
               //Paso 6
               suma = suma + arreglo[x];
          }
     }
}
```

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

PASO 7:

Este método lo llamamos dentro del método main.

```
public static void main(String[] args) {
    //Paso 7
    System.out.println("La suma es: " + suma());
}
```

Sumar dentro de un arreglo

Construir un programa que sume valores que estén entre 1 y 10 dentro de un arreglo.

Finalmente el ejercicio quedaría de la siguiente manera:

```
public static void main(String[] args) {
     System.out.println("La suma es: " + suma());
     }
     public static int suma() {
          int suma = 0;
          int[] arreglo = { 1, 5, 11, 33, 4, 6, 7, 44, 6, 1, -1 };
          for (int x = 0; x < arreglo.length; x++) {
               if (arreglo[x] >= 1 && arreglo[x] <= 10) {
               suma = suma + arreglo[x];
          }
     }
     return suma;
}
```

Ejercicio propuesto

"Construir un programa que sume valores que estén entre 1 y 5 dentro de un arreglo"

Según lo aprendido, ¿cuál es la importancia de un arreglo?

De los siguientes códigos, identifica:

La alternativa correcta para declarar un arreglo de tipo entero de capacidad máxima 2

- `int valor = 2;`
- `int [ ] valor = new int [3] ;`
- `int [ ] valor = new int [2];`
- `response: 'valor = int [ ] 2;`

Próxima sesión...

- *Comprender la documentación de la clase ArrayList para hacer uso de sus métodos y codificar de manera rápida.*
- *Aplicar operaciones de un array dinámico para "agregar", "eliminar", "ordenar" y "contar" para conocer los métodos esenciales de un ArrayList.*

{desafío}
latam_

Academia de
talentos digitales