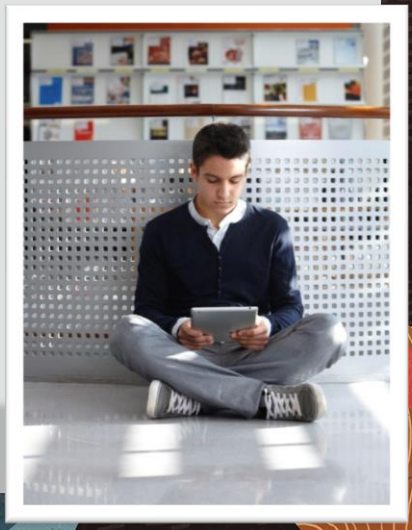
ORACLE

Academy

Java Fundamentals

6-1:
Arreglas



ORACLE
Academy

Objetivos

- Esta lección abarca los siguientes temas:
  - Escribir una arreglo unidimensional en un programa Java usando tipos de datos primitivos
  - Escribir una arreglo unidimensional en un programa Java usando tipos de referencia (objetos)
  - Escribir una arreglo bidimensional en un programa Java usando tipos de datos primitivos
  - Escribir una arreglo bidimensional en un programa Java usando tipos de referencia (objetos)
  - Declarar, inicializar y desviar una arreglo

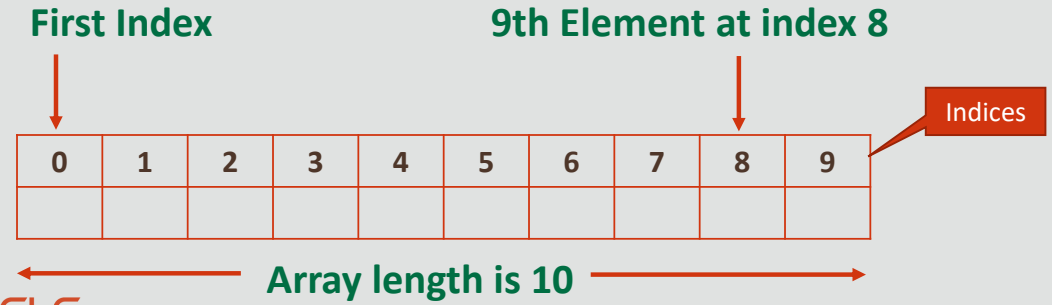
ORACLE Academy

Descripción general

- Esta lección abarca los siguientes temas:
  - Describir la inicialización de la arreglo
  - Distinguir entre el método length() de String y el valor de longitud de la arreglo
  - Reescribir un programa Java para almacenar enteros en una arreglo, realizar un cálculo matemático y mostrar el resultado
  - Usar sintaxis alternativas de declaración de arreglas

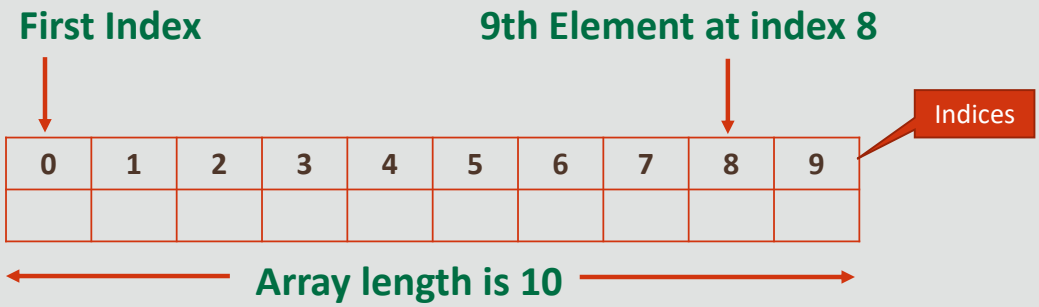
¿Qué es una arreglo?

- Una arreglo es una colección de valores del mismo tipo de datos almacenados en un objeto contenedor
- Puede ser cualquier número de valores
- Cuando se declara la arreglo se establece su longitud
- Una vez que se declara la arreglo, se establece su tamaño



Elementos de la arreglo

- Cada valor se denomina elemento
- Se accede a cada elemento mediante un índice
- El índice debe ser un entero
- El índice siempre comienza en 0



ORACLE Academy

Tipos de datos de las arreglas

- Las arreglas pueden contener cualquier tipo de datos, como los siguientes:
  - Primitivos
  - Objetos predefinidos, como Strings de las API de Java
  - Objetos definidos por programadores, como las instancias de la clase que usted cree

Declaración de una arreglo

- La declaración de una arreglo consta de tres partes:
  - Tipo de datos
  - Nombre de la variable
  - Tamaño de la arreglo

Ejemplos de declaración de una arreglo

- La declaración de una arreglo se puede realizar en una o dos líneas
- Ambos ejemplos a continuación son equivalentes a declaraciones de arreglas

```
data_type[] variable_name;
variable_name = new data_type[size];
//declare an array using two lines of code
```

```
data_type[] variable_name = new data_type[size];
//declare an array using one line of code
```

Ejemplo de declaración de una arreglo

- Un amigo le trae un ramo de seis flores diferentes
- Desea escribir un programa para almacenar cada tipo de flor en el ramo
- Los segmentos del código a continuación son formas idénticas de declarar una arreglo de Strings denominada myBouquet
- El tamaño se establece para seis elementos, de modo que puede almacenar cada tipo de flor en su ramo

```
String[] myBouquet;
myBouquet = new String[6];
```

```
String[] myBouquet = new String[6];
```

Explicación del ejemplo de arreglo myBouquet

```
String[] myBouquet;
myBouquet = new String[6];
```

```
String[] myBouquet = new String[6];
```

El tamaño de la arreglo es de 6 elementos, uno para cada flor

El tipo de datos almacena los tipos de flores como Strings

El nombre de la arreglo es myBouquet

ORACLE Academy

Identificar los componentes de la declaración de una arreglo

- Identificar los tres componentes de la declaración de una arreglo para cada una de estas arreglas de tipos de datos primitivos

```
1.   int[] myArray;
     myArray = new int[20];

2.   char[] sentence = new char[100];

3.   double[] teamPoints = new double[5];
```

	Data Type	Name	Size
1			
2			
3			

ORACLE Academy

Sintaxis alternativa de la declaración de arreglas

- Una manera alternativa de declarar las arreglas es posicionar los corchetes utilizados para declarar una arreglo a cada lado del identificador de la arreglo
- Sintaxis presentada:

```
int[] primeNumbers;
int[] evenNumbers;
double[][] prices;
String[] words;
Point[] coordinates;
Rectangle[][] blocks;
```

Sintaxis alternativa:

```
int primeNumbers[];
int evenNumbers[];
double prices[][];
String words[];
Point coordinates[];
Rectangle blocks[][];
```


Los programadores de Java utilizan rara vez la sintaxis alternativa.

Sintaxis alternativa de declaración de arreglas

- Cualquiera de las declaraciones funciona
  - La sintaxis alternativa es similar a la que se utiliza en lenguaje C
  - La sintaxis utilizada para presentar arreglas en realidad se lee de manera más clara
  - La primera línea se leerá “arreglo de enteros primeNumbers”

- Syntax introduced:

```
int[] primeNumbers;
int[] evenNumbers;
double[][] prices;
String[] words;
Point[] coordinates;
Rectangle[][] blocks;
```

Alternate syntax:

```
int primeNumbers[];
int evenNumbers[];
double prices[][];
String words[];
Point coordinates[];
Rectangle blocks[][];
```

Inicialización de una arreglo

- Una vez que declara una arreglo debe inicializarla para establecer los valores de los índices especificados
- La inicialización de una arreglo consta de tres componentes:
  - Nombre de la variable
  - Índice
  - Valor

Formas de inicialización de una arreglo

- Una arreglo se puede inicializar de dos maneras diferentes:
  - Declare la arreglo, luego inicialice cada elemento
  - Realice la declaración y la inicialización en el mismo paso
- Ejemplo de declaración e inicialización en dos pasos:

```
data_type[] variable_name = new data_type[size];
variable_name[index] = value;  //repeat for each index
```

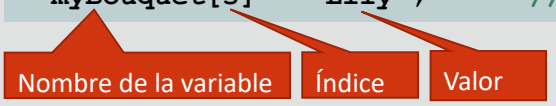
- Ejemplo de declaración e inicialización en un paso:

```
data_type[] variable_name = {val1, val2, …};
```

Ejemplo 1 de inicialización de una arreglo

- Recuerde la declaración de la arreglo del String myBouquet
- Una vez declarada la arreglo, el código a continuación inicializa los elementos y almacena los tipos de flores
- El índice de una arreglo comienza en 0, de modo que el primer elemento se agrega al índice 0

```
String[] myBouquet = new String[6];//previous declaration
 myBouquet[0] = "Rose";       //Store "Rose" as the first element
 myBouquet[1] = "Sunflower"; //Store "Sunflower" as the second
 myBouquet[2] = "Daisy";      //and so on
 myBouquet[3] = "Dandelion";
 myBouquet[4] = "Violet";
 myBouquet[5] = "Lily";       //"Lily" is the last (sixth) element
```



Ejemplo 1 de inicialización de una arreglo

- La arreglo de tipos de flores se verá de la siguiente manera:

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Value:	Rose	Sunflower	Daisy	Dandelion	Violet	Lily

ORACLE Academy

Ejemplo 2 de inicialización de una arreglo

- La arreglo myBouquet también se puede declarar e inicializar utilizando la segunda notación de la siguiente manera:

```
String[] myBouquet = {"Rose", "Sunflower", "Eucalyptus",
                      "Dandelion", "Violet", "Lily"};
```

- Observe que al usar este método no se especifica el tamaño, pero se asigna un tamaño en base a la cantidad de elementos de la lista entre corchetes "{}"

A menudo, cada elemento se coloca en una línea automáticamente de modo que el código es más obvio.

Primera notación frente a la segunda notación

- Aparentemente la codificación de la segunda notación es mucho más corta y sencilla
- ¿Por qué necesitará la primera notación?

Primera notación frente a la segunda notación

- Considere tomar cinco números de un usuario y almacenarlos en una arreglo denominada myArray
- Necesitará declarar la arreglo, luego inicializar los elementos de a uno por vez, a medida que el usuario ingrese cada número

```
int[] myArray = new int[5]; //Declare the array of size 5
MyArray[0] = 7;             //The user entered 7
MyArray[1] = 24;            //The user entered 24
MyArray[2] = 352;           //The user entered 352
MyArray[3] = 2;             //The user entered 2
MyArray[4] = 37;            //The user entered 37
```

Primera notación frente a la segunda notación

- Es posible que la segunda notación aparentemente refleje una manera más sencilla de inicializar la arreglo, si ya conoce el contenido de la arreglo al declararlo

```
int[] myArray = {7, 24, 352, 2, 37};
```

Representación de arreglas

- Cuando se inicialan las arreglas pero no se inicializan, los elementos reciben el valor predeterminado relacionado con el tipo de datos
- Por ejemplo, el valor predeterminado para los tipos de datos numéricos, como int (entero), es 0
- El valor predeterminado para tipos de objeto, como String es “” (null)
  - Cuando se declara la arreglo, la representación en la tabla es la siguiente

```
int[] myArray = new int[5];
```

Índice:	0	1	2	3	4
Valor:	0	0	0	0	0

ORACLE Academy

Representación de arreglas actualizada

- Una vez que comienza a inicializar elementos, la arreglo se actualiza
- La nueva representación en la tabla es la siguiente

```
myArray[0] = 32;
myArray[3] = 27;
```

Índice:	0	1	2	3	4
Valor:	32	0	0	27	0

Tipos de objetos de arreglas

- Las arreglas no están restringidas al almacenamiento de tipos de datos primitivos
- Pueden almacenar cualquier tipo de objetos, incluso los tipos que defina
- Por ejemplo, si existe una clase de flor:
  - Las flores se podrán almacenar en la arreglo en lugar de almacenar el tipo de flor como String
  - Teniendo en cuenta que sabemos qué flores se pueden incluir, la segunda notación se puede usar para inicializar myBouquet de seis flores

```
Flower[] myBouquet = {new Flower("Rose"), new Flower("Sunflower"),
                      new Flower("Daisy"), new Flower("Dandelion"),
                      new Flower("Violet"), new Flower("Lily")};
```

Acceso a la longitud de la arreglo

- Con cada declaración de una arreglo, puede definir el tamaño o la longitud de la arreglo
- La longitud se almacena como variable de una instancia para ese objeto y se puede acceder usando la notación arrayName.length
- Esta técnica es útil en el siguiente ejemplo:
  - Establecer una arreglo cuyo tamaño se base en el ingreso del usuario
  - Ingresar un segmento de código donde el ingreso del usuario ya no se encuentra en el ámbito
  - Necesitará acceder a la longitud de la variable de la instancia para esa arreglo
  - En resumen, arr.length devuelve la longitud de la arreglo, arr

Con Strings, existía un método length(). Con Arreglas, solo existe el método length sin los paréntesis ().

Recorrido de iteración en una arreglo

- Para iterar o desviar una arreglo significa avanzar en cada elemento de la arreglo por número de índice
- El recorrido en iteración de una arreglo es útil cuando:
  - desea acceder a cada elemento de una arreglo en orden
  - desea inicializar todos los elementos de una arreglo en el mismo valor
- Usar .length al recorrer en iteración en lugar del valor entero ingresado al declarar la arreglo
- De este modo se garantizará que no reciba un índice del error de los límites

ORACLE Academy

También podría iterar a través de una arreglo e inicializar cada elemento en un valor distinto. Podría pedir al usuario los valores o podría leerlos desde un archivo.

Ejemplo de recorrido de iteración en una arreglo

- Para inicializar una arreglo de enteros denominada allTwos de modo que cada elemento sea 2 utilice un ciclo for, como se indica a continuación
- La primera línea del código corresponde a la declaración de la arreglo
- Declara una arreglo denominada allTwos con un tamaño de 10

Ejemplo de recorrido de iteración en una arreglo

- El ciclo for recorre en iteración los índices y para cada índice en la arreglo, el valor de ese índice se establece en 2

```
int[] allTwos = new int[10];
for(int index = 0; index < allTwos.length; index++){
   allTwos[index] = 2;
}//end for
```

Observe de qué manera se accede a la longitud de la arreglo para no salir de los límites del índice de la arreglo

Cuándo es útil la iteración

- En el ejemplo de la flor, la iteración le ayuda si desea imprimir los tipos de flores almacenados en la arreglo myBouquet

```
String[] myBouquet = {"Rose", "Sunflower", "Daisy", "Dandelion",
                      "Violet", "Lily"};
```

- Utilice un ciclo for para recorrer esta arreglo
- El contador inicializado en el ciclo for se puede utilizar para aumentar los índices, como se indica a continuación
  - ¿Qué se visualiza como resultado de este código?

```
//remember that the index range is 0 to 5 for an array of size 6
  for(int index = 0; index < myBouquet.length; index++){
          System.out.println(myBouquet[index]);
```

ciclo for-each

- Java ofrece un ciclo for-each, una alternativa del uso del contador inicializado para iterar en una arreglo
  - Cuando se utiliza para acceder a los elementos de una arreglo, el ciclo for- each funciona de la misma manera que en el ciclo for, pero se implementa de forma más sencilla
  - Si reemplazamos el código del ciclo for de nuestro ejemplo anterior con el código siguiente, obtenemos el mismo resultado

```
//remember that the index range is 0 to 5 for an array //of size 6
for (String myFlower : myBouquet)
{
    System.out.println(myFlower);
}//end for
```


El ciclo for-each tiene una sintaxis más simple que el ciclo for para iterar a través de una arreglo pero no tiene una variable de control del ciclo. Por lo tanto, si por ejemplo desea imprimir cada elemento con el número de elemento que aparece delante, el ciclo for-each no sería la mejor opción.

ciclo for-each

- El ciclo for-each tiene acceso (de a uno por vez) y devuelve todos los elementos de una arreglo
  - Los cambios de los elementos de la arreglo no se pueden hacer usando un ciclo for-each
  - Si reemplazamos el código del ciclo for de nuestro ejemplo anterior con el código siguiente, obtenemos el mismo resultado
  - El ejemplo a continuación imprimirá la longitud de cada string en la arreglo myBouquet

```
//remember that the index range is 0 to 5 for an array //of size 6
for (String myFlower : myBouquet)
{
    System.out.println(myFlower);
}//end for
```

Ejemplo de ciclo for-each

- Ambas implementaciones del código siguiente mostrarán información relacionada con los elementos de la arreglo

```
public class Bouquet {
   public static void main(String[] args){
      String[] myBouquet = {"Rose","Sunflower","Daisy",
                            "Dandelion","Violet", "Lily"};

      //use a for loop to iterate through the array
      for(int index = 0; index < myBouquet.length; index++){
           System.out.println(myBouquet[index]);
      }//end for
      //use a for each to iterate through the array
      for (String myFlower : myBouquet){
        System.out.println(myFlower);
      }//end for
   }//end main
}//end class Bouquet
```

Qué sabemos acerca de las arreglas

- Qué sabemos acerca de las arreglas:
  - Las arreglas son un tipo de objeto que puede almacenar cualquier tipo primitivo o de objeto
  - Por lo tanto, las arreglas pueden almacenar arreglas
  - El concepto de almacenamiento de una arreglo de arreglas se denomina arreglo bidimensional

Arreglas bidimensionales

- Una arreglo bidimensional, denominada “arreglo de arreglas”, es una arreglo que almacena otras arreglas
- La cantidad de arreglas contenidas en la arreglo se define en la declaración
- La cantidad de elementos en cada arreglo interna se define también en el momento de la declaración

No existe un límite en Java en cuanto al número de dimensiones que puede tener una arreglo. La limitación es la cantidad de memoria que tiene el equipo para almacenar elementos de la arreglo (y su estado).

Ejemplo de arreglo bidimensional

- Este ejemplo muestra una arreglo de dos arreglas con tres elementos en cada arreglo
- Una arreglo bidimensional se puede visualizar como una tabla con filas y columnas
- El ejemplo a continuación tiene dos filas y tres columnas

```
int[][] nums = { {14,51,16}, {12,73,87} };
```

Declaración de una arreglo bidimensional

- Componentes de arreglas bidimensionales:
  - Tipo de datos
  - Nombre de la variable
  - Tamaño de la arreglo
- Para declarar una arreglo bidimensional, use cualquiera de los ejemplos de sintaxis que se indican a continuación

```
data_type[][] variable_name;
variable_name = new data_type[size1][size2];
//declare it using two lines of code
```

```
data_type[][] variable_name = new data_type[size1][size2];
//declare it using one line of code
```

Ejemplo 1 de declaración de una arreglo bidimensional

- Identificar los tres componentes en los siguientes ejemplos de tipos de datos primitivos

```
int[][] myArray;
myArray = new int[2][3];
char[][] sentence = new char[10][10];
```

Ejemplo 1 de declaración de una arreglo bidimensional

- Al declarar una arreglo bidimensional:
  - el primer número entre corchetes [2] corresponde al número de arreglas que tiene el contenedor (filas)
  - El segundo número entre corchetes [3] corresponde al número de elementos en cada una de estas arreglas (columnas)
- Otra manera de declarar myArray, puede ser la siguiente:

```
int[][] myArray = { {0,0,0}, {0,0,0} };
```

Ejemplo 2 de declaración de una arreglo bidimensional

- Su amigo le trajo tres flores de cada tipo y cada una de las flores es de diferente color
- Con una arreglo unidimensional resulta tedioso realizar el seguimiento de estos datos
- Una arreglo bidimensional le permite almacenar seis arreglas, una por cada tipo de flor y permite que cada arreglo almacene los colores de cada una de las tres flores
- El código siguiente declara una arreglo que contiene seis arreglas, cada una con una longitud de tres arreglas:

```
String[][] myBouquet = new String[6][3];
```

Inicialización de una arreglo bidimensional

- Las arreglas bidimensionales, al igual que las arreglas unidimensionales, se pueden inicializar usando dos métodos diferentes

Un ciclo dentro de otro ciclo se conoce como anidamiento. No existe un límite en Java en cuanto al número de niveles de profundidad en los que se pueden anidar los ciclos.

Inicialización de una arreglo bidimensional

- Método 1:
  - i es el índice de la arreglo interna (fila) y j es el índice del elemento en esa arreglo (columna) que se inicializa

```
public class TwoDTester{
  public static void main(String[] args){
    Scanner in = new Scanner(System.in);
    int[][] nums = new int[3][2];

    for(int i = 0; i < nums.length; i++){
      for(int j = 0; j < nums[i].length; j++){
        System.out.println("Enter a value for row " + i + ", column " + j);
        nums[i][j] = in.nextInt();
      }//end for
    }//end for
  }//end main
}//end class TwoDTester
```


Un ciclo dentro de otro ciclo se conoce como anidamiento. No existe un límite en Java en cuanto al número de niveles de profundidad en los que se pueden anidar los ciclos.

Inicialización de una arreglo bidimensional

- Método 2: Declara e inicializa todas las arreglas y todos los elementos en estas arreglas en la misma línea del código
- Sin embargo, debe conocer los valores que desea que contenga la arreglo para inicializar la arreglo en el mismo momento en que se declara

```
public class TwoDTester2{
   public static void main(String[] args){
      int[][] nums = {{2,3,7},{15, 98, 2}};
   }//end main
}//end class TwoDTester2
```

Ejemplo de inicialización de una arreglo bidimensional

- Método 1:

```
int[][] myArray = new int[3][2];
MyArray[0][0] = 7;
MyArray[0][1] = 24;
MyArray[1][0] = 352;
MyArray[1][1] = 2;
MyArray[2][0] = 37;
MyArray[2][1] = 65;
```

Índice de la arreglo interna

Índice del elemento de la arreglo interna

- Método 2:

```
int[][] myArray = new int[][] {{7, 24}, {352, 2}, {37, 65}};
```

Uso de la segunda notación para inicializar la arreglo

- Dado que ya conocemos el contenido de las arreglas para nuestro ramo, use la segunda notación para inicializar la arreglo
- Recuerde que el orden es “rosa”, “girasol”, “margarita”, “diente de león”, “violeta” y, luego, “lirio”
- Estas flores estarán representadas por los índices 0, 1, 2, 3, 4 y 5

```
String[][] myBouquet = {{"Red", "Peach", "Yellow"},
                        {"Yellow", "White", "Blue"},
                        {"Green", "Blue", "Purple"},
                        {"White", "White", "White"},
                        {"Purple", "Pink", "Violet"},
                        {"Pink", "Orange", "White"}};
```

Representación de arreglo bidimensional

- La arreglo bidimensional se representa de la siguiente manera

Índice	0	1	2
0 (rosa)	Rojo	Durazno	Amarillo
1 (girasol)	Amarillo	Blanco	Azul
2 (margarita)	Verde	Azul	Púrpura
3 (diente de león)	Blanco	Blanco	Blanco
4 (violeta)	Púrpura	Rosa	Violeta
5 (lirio)	Rosa	Naranja	Blanco

Visualización de una arreglo bidimensional

- Otra manera de visualizar una arreglo bidimensional es al alinear los valores en el momento de la inicialización
- Esta técnica ayuda a realizar el seguimiento de la manera en que se organizan los datos

Este es un gran ejemplo de por qué cada fila debe aparecer automáticamente en una línea. Como se indicó al principio, no es fácil ver inmediatamente lo que está sucediendo.

Visualización de una arreglo bidimensional

- En los ejemplos de código siguientes , observe cómo la segunda forma es más fácil de visualizar

```
String[][] myBouquet = {{"Red", "Peach", "Yellow"}, {"Yellow",
"White", "Blue"}, {"Green", "Blue", "Purple"}, {"White", "White",
"White"}, {"Purple", "Pink", "Violet"}, {"Pink", "Orange",
"White"}};
```

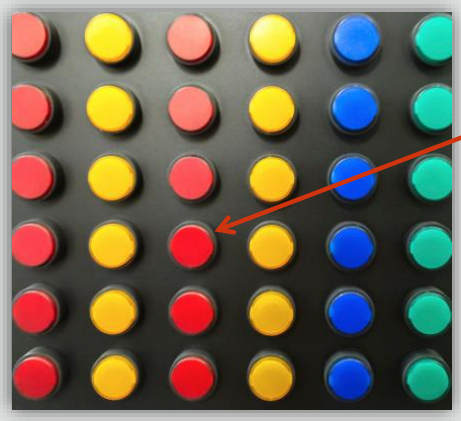
```
String[][] myBouquet = {{"Red", "Peach", "Yellow"},
                        {"Yellow", "White", "Blue"},
                        {"Green", "Blue", "Purple"},
                        {"White", "White", "White"},
                        {"Purple", "Pink", "Violet"},
                        {"Pink", "Orange", "White"}};
```

ORACLE Academy

Este es un gran ejemplo de por qué cada fila debe aparecer automáticamente en una línea. Como se indicó al principio, no es fácil ver inmediatamente lo que está sucediendo.

Acceso a los datos en una arreglo bidimensional

- Para acceder a los datos en una arreglo bidimensional, debe conocer:
  - El índice de la arreglo para el acceso
  - El índice del contenido en esa arreglo para el acceso



Ejemplo de acceso a los datos en una arreglo bidimensional

- Por ejemplo, para acceder al color de la rosa en myBouquet, use el índice de la arreglo rosa (0) y el índice del primer color (0)

```
//Previously initialized array
String[][] myBouquet = String[][] {
                     {"Red", "Peach", "Yellow"},
                     {"Yellow", "White", "Blue"},
                     {"Green", "Blue", "Purple"},
                     {"White", "White", "White"},
                     {"Purple", "Pink", "Violet"},
                     {"Pink", "Orange", "White"}
                     };
String rose1 = myBouquet[0][0]; //accesses first element
                                // of first array
```

Ejemplo de acceso a los datos en una arreglo bidimensional

- Revisión de la inicialización de la arreglo anterior

```
//Previously initialized array
String[][] myBouquet = String[][] {
                {"Red", "Peach", "Yellow"},
                {"Yellow", "White", "Blue"},
                {"Green", "Blue", "Purple"},
                {"White", "White", "White"},
                {"Purple", "Pink", "Violet"},
                {"Pink", "Orange", "White"}};
```

- ¿Qué se visualizará en la pantalla de la consola después de ejecutar el siguiente código?

```
System.out.println(myBouquet[0][1] + " Rose.";
System.out.println(myBouquet[5][2] + " Lilly.";
```

Solución del ejemplo de acceso a los datos en una arreglo bidimensional

- ¿Qué se visualizará en la pantalla de la consola después de ejecutar el siguiente código?

```
System.out.println(myBouquet[0][1] + " Rose.";
System.out.println(myBouquet[5][2] + " Lilly.";
```

- Solución:
  - Peach Rose
  - White Lilly

Acceso a la longitud de arreglas bidimensionales

- La longitud es una variable de la instancia definida por el tamaño de cada arreglo declarada
- Las arreglas bidimensionales tienen dos longitudes diferentes:
  - Longitud de la arreglo externa
  - Longitud de las arreglas internas

La longitud de la arreglo exterior es la misma que el número de filas. La longitud de las Arreglas internas es la misma que la cantidad de columnas.

Longitud de las arreglas externas e internas

- Se accede a la longitud de la arreglo externa, que es la longitud que describe la cantidad de arreglas contenidas (filas), como a una arreglo típica

```
int numArrays = arrayName.length;
```

- Se accede a la longitud de las arreglas internas, que es la longitud que describe la cantidad de elementos que contiene cada arreglo (columnas), usando la siguiente notación
- Los corchetes [ ] le indican al programa que están accediendo a la longitud de las arreglas internas y la fila indica cuál es la arreglo

```
int numElementsInEach = arrayName[row].length;
```

Ejemplo de longitud de las arreglas externas e internas

- Revisión de este código

```
String[] one = new String[7];
int[][] two = new int[5][3];
double[][] three = new double[3][2];
People[] four = new People[5];\
```

- ¿Cuál de los siguientes enunciados no es verdadero? ¿Puede identificar cuál incluye una sintaxis inadecuada para acceder a la longitud?

```
one.length == 7;
two.length == 3;
three.length == 3;
two[0].length == 3;
three[0].length == 2;
four[0].length == 5;
```

Solución del ejemplo de longitud de las arreglas externas e internas

- Solución:
  - `two.length == 3;` **es FALSO**
  - `four[0].length == 5;` **no es una sintaxis válida**

Tipos de objetos de arreglas bidimensionales

- Al igual que las arreglas unidimensionales, las arreglas bidimensionales pueden almacenar cualquier tipo de objeto




ORACLE Academy

Tipos de objetos de arreglas bidimensionales

- ¿De qué manera podemos organizar a los estudiantes en un aula en tres grupos, con cinco estudiantes por grupo como una arreglo?
- Respuesta: Declare una arreglo bidimensional para contener las tres arreglas, una para cada grupo
- Cada arreglo almacena cinco estudiantes
- Se ha definido una clase denominada estudiante y tiene un constructor que toma el primer nombre del estudiante
- El siguiente código declara la arreglo que almacena el grupo de estudiantes

```
Student[][] groups = new Student[3][5];
```

Ubicación de los estudiantes en la arreglo bidimensional

- Dada una arreglo de String que contiene los nombres de 15 estudiantes en los grupos, ¿de qué manera podría realizar las siguientes tareas?
  - Iterar en la arreglo existente
  - Crear un nuevo estudiante para cada nombre
  - Colocar a cada estudiante en uno de los tres grupos

Ubicación de los estudiantes en la arreglo bidimensional

- Usted ya sabe de qué manera:
  - Iterar en una arreglo utilizando un ciclo for
  - Crear un nuevo estudiante para cada nombre en el ciclo
  - Ubicar a los estudiantes en la arreglo bidimensional usando un ciclo anidado for

ciclos anidados for

- Un ciclo anidado for:
  - Es un ciclo for dentro de otro ciclo for
  - Se puede utilizar para iterar en arreglas bidimensionales usando el ciclo for externo como el índice de las arreglas (filas) y el contador interno del ciclo for, como el índice de los elementos de cada arreglo (columnas)
- Considere la siguiente declaración de una arreglo bidimensional:

```
Student[][] groups = new Student[3][5];
```

ciclos anidados for

- Para iterar en la arreglo e inicializar a cada estudiante como “Temp”, use los ciclos anidados for, como se indica a continuación

```
//i keeps track of the rows
for(int i = 0; i < groups.length; i++){
   //j keeps track of the columns
   for(int j = 0; j < groups[i].length; j++){
      groups[i][j] = new Student("Temp");
   }//end for
}//end for
```

Completar las tareas usando los ciclos anidados for

- ¿De qué manera podríamos completar las tres tareas?
  - Iterar en la arreglo existente
  - Crear un nuevo estudiante para cada nombre
  - Colocar a cada estudiante en uno de los tres grupos

Completar las tareas usando los ciclos anidados for

```
String[] studentNames;
//Assume studentNames is initialized with 15 names

Student[][] groups = new Student[3][5];//Declare your array
   int x = 0;
   for(int i = 0; i < groups.length; i++){
      for(int j = 0; j < groups[i].length; j++){
         String name = studentNames[x];
         groups[i][j] = new Student(name);
         x++;
      }//End inner for loop
  }//End outer for loop
```

ORACLE Academy

Para bloques anidados, se recomienda poner un comentario después de la llave de cierre de los ciclos anidados (/// End inner for loop) para que podamos identificar fácilmente dónde termina cada bloque de código.

Argumentos de línea de comandos

- Una arreglo siempre ha sido parte del método principal
- El args de la arreglo de String siempre se considera un parámetro para el método principal

```
public static void main(String[] args)
```


Los argumentos de la línea de comandos se utilizaban bastante más antes de que el uso de las GUI se extendiera.

Argumentos de línea de comandos: Agregado de argumentos adicionales

- Si ejecuta un programa Java desde un entorno de una línea de comandos, puede escribir argumentos adicionales, de la siguiente manera:

```
java Test apples peaches pumpkin pie
```

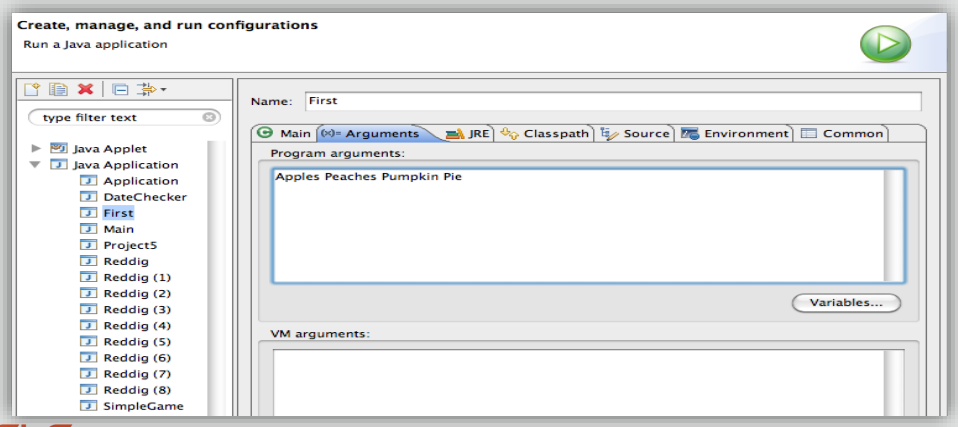
- El término Java indica el entorno de la línea de comandos para utilizar el JVM para ejecutar la prueba del programa

Argumentos de línea de comandos: Agregado de argumentos adicionales

- Los args de la arreglo se completan automáticamente con los Strings: manzanas, duraznos, calabazas y pastel
- Al usar Eclipse, evitamos el entorno de la línea de comandos
- Sin embargo, podemos usar argumentos de la línea de comandos en Eclipse

Uso de los argumentos de la línea de comandos en Eclipse

- Acceda a la pestaña de argumentos para su programa, diríjase al menú Ejecutar y elija Abrir diálogo ejecutar
- Haga clic en la pestaña Argumentos y escriba los Strings: manzanas, duraznos, calabazas y pastel

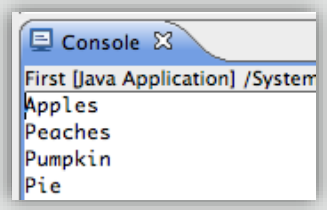


Coloque "comillas dobles" alrededor de los argumentos de varias palabras. Por ejemplo, rojo "azul claro" verde. Estos son tres argumentos. Sin las comillas, se considerarían cuatro argumentos.

Uso de los argumentos de la línea de comandos en Eclipse

- Haga clic en Ejecutar, para este programa
- Todos los Strings que se escribieron en la pestaña Argumentos se imprimirán en la consola

```
public class TestArgs {
   public static void main(String[] args){
      for(int i=0;i<args.length;i++){
         System.out.println(args[i]);
      }//end for
   }//end main
}//end class TestArgs
```



ORACLE Academy

JF 6-1
Arreglas

Terminología

- Los términos clave usados en esta lección son los siguientes:
  - Algoritmo
  - arreglo
  - arreglo de arreglas
  - Argumento de la línea de comandos
  - Índice
  - Iterar
  - ciclo anidado for
  - arreglo unidimensional
  - Desviar
  - arreglo bidimensional

Resumen

- En esta lección, habrá aprendido a:
  - Escribir una arreglo unidimensional en un programa Java usando tipos de datos primitivos
  - Escribir una arreglo unidimensional en un programa Java usando tipos de referencia (objetos)
  - Escribir una arreglo bidimensional en un programa Java usando tipos de datos primitivos
  - Escribir una arreglo bidimensional en un programa Java usando tipos de referencia (objetos)
  - Declarar, inicializar y desviar una arreglo

Resumen

- En esta lección, habrá aprendido a:
  - Describir la inicialización de la arreglo
  - Distinguir entre el método length() de String y el valor de longitud de la arreglo
  - Reescribir un programa Java para almacenar enteros en una arreglo, realizar un cálculo matemático y mostrar el resultado
  - Usar sintaxis alternativas de declaración de arreglas


ORACLE
Academy