 



Conceptos fundamentales de Java 6-1: Arreglas

Actividades prácticas

# Objetivos de la lección:

* Realización de un array unidimensional en un programa Java usando tipos de datos primitivos
* Realización de un array unidimensional en un programa Java usando tipos de referencia (Objeto)
* Realización de un array bidimensional en un programa Java usando tipos de datos primitivos
* Realización de un array bidimensional en un programa Java usando tipos de referencia (Objeto)
* Declaración de un array, inicialización de un array y atravesar (traverse) el array
* Descripción de la inicialización de array
* Distinción entre el método de longitud de cadena length() y el valor de longitud del array
* Reescribir un programa Java para guardar enteros en un array, realizar un cálculo matemático y mostrar el resultado
* Uso de sintaxis alternativa de declaración de array

# Vocabulario:

Identifique el término correspondiente a cada una de las definiciones a continuación.

|  |  |
| --- | --- |
| iteracion | Acto de pasar por un array. |
| Array o arreglo | Estructura que almacena valores múltiples del mismo tipo de datos. |
| matriz | Array de dos dimensiones. |
| Índice | Número entero que identifica la ubicación de un valor en un array. |
| Paso de parámetros por referencia | Capacidad de transmitir datos hacia la función principal y acceder a ellos como elemento del array. |
| Algoritmo | Procedimiento informático lógico que, si se aplica correctamente, garantiza la solución de un problema. |
| Matriz multidimensional o arreglo bidimensional | Un array de arrays, similar a una tabla, arreglo u hoja de cálculo. |
| Adinacion de bluces for | A para ciclo dentro de "a" para ciclo |
| Variable | Objeto con nombre utilizado para almacenar más de un valor. |

# Inténtelo/resuélvalo:

1. Declare una calificación de nombre de array de una dimensión del tipo int que pueda contener 9 valores.

R// Int [] calificación = new Int [9];

1. Declare un precio con nombre de array de 2 dimensiones de tipo float que tenga 10 filas y 3 columnas.

R// Float [][] precio = new float[10],[3];

1. Declare e inicialice un array de 2 dimensiones denominado arreglo de un tipo largo que tenga 4 filas y 3 columnas para que todos los valores estén configurados en 5.

R// Long [][] arreglo = {{5,5,5}, {5,5,5}, {5,5,5}, {5,5,5}};

1. Declare e inicialice un array de bytes de una dimensión denominado valores de tamaño 10 de manera que todas las entradas contengan 1.

R// byte [] valores = new byte [10];

Arrays. fill (valores) (byte) 1);

1. Sin escribir el código, determine el resultado del siguiente programa.

int num[] = {7,7,6,6,5,5,4,4};

for(int i = 0; i < 8; i = i + 2)

System.out.print(num[i]);

R// Es la impresión de los elementos del índice del array par “num” , que son 7,6,5,4 dando un valor 7654

1. Sin escribir el código, determine el resultado del siguiente programa. int[][] num = {{3,3,3},{2,2,2}};

int[] array = {4,3,2}; for(int i = 0; i < 3; i++){

num[1][i] = num[0][i]+array[i];

}

for(int i = 0; i < 2; i++){

for(int j = 0; j < 3; j++){

System.out.print(num[i][j]);

}

System.out.println();

}

R// El programa crea una matriz de “num” 2x3 y un array de longitud 3 Luego, se recorre el array "array" y se suma cada elemento con el correspondiente elemento de la primera fila de la matriz "num". El resultado de cada suma se asigna al elemento correspondiente de la segunda fila de la matriz "num". Da un resultado de 333 y 765

1. En una clase determinada, existen 5 pruebas que valen 100 puntos cada una. Escriba un programa que tome los 5 resultados de las pruebas para el usuario, guarde los resultados de las pruebas en un array y, luego, calcule el promedio de los estudiantes.

1. En la clase de Álgebra, aprendemos sobre las arreglas. Aprendemos a sumar, restar y multiplicar arreglas de 2x2 y 3x3. A continuación presentamos ejemplos de la clase de Álgebra con sus respuestas:

�3 4�+� 1 0�=�4 4�

5 6 −2 3 3 9

�3 4�-� 1 0�=�2 4�

5 6 −2 3 7 3

�3 4�\*� 1 0�= −5 12

5 6 −2 3 � �

−7 18

Es casi aparente cómo sumar. Debemos sumar la primera posición a la primera arreglo con la primera posición en la segunda arreglo. Continuamos con las posiciones correspondientes para obtener la respuesta. La resta sigue este mismo método de posición. La multiplicación de arreglas parece ser confusa, ya que no sigue el método de posiciones utilizado en la suma y resta.

Para obtener la respuesta, se toma la fila de la primera arreglo y la columna de la segunda arreglo y se multiplican los respectivos valores y, luego, se toma la suma de los resultados.

La respuesta anterior se obtuvo de la siguiente manera:

3(1)+4(-2)=-5 3(0)+4(3)=12

5(1)+6(-2)=-7 5(0)+6(3)=18

Escriba un programa que tome dos arreglas y, luego, permita al usuario elegir sumar, restar o multiplicar y mostrar la respuesta. El programa mostrará el siguiente menú:

* 1. Enter Arreglo A
  2. Enter Arreglo B
  3. Display A + B
  4. Display A - B
  5. Display A \* B
  6. Exit

El programa debe mostrar un ciclo y permitir al usuario continuar eligiendo diferentes opciones hasta que eligen salir. El programa bien escrito colocará en módulos el proceso en métodos diferentes.

1. Escriba el código y prueba. Añadir las siguientes características.
   1. Agregue un método para mezclar a la clase Deck. Llame al método desde la clase Main (principal) para verificar que el mazo esté verdaderamente mezclado.
   2. Agregue una clase Hand (mano) que contenga un array de 5 referencias de cartas. Haga al programa repartir dos cartas por mano y mostrarlas al usuario. Indique al usuario cuántos puntos tiene y pregunte a dicho usuario si quiere otra carta o no. Continúe y permítale al jugador agregar cartas hasta llegar a 5 cartas o hasta que el total sea mayor a 21.
   3. Ajuste la clase Card (carta) para permitir a los Ases que cuenten como 11 para comenzar. Si la clase Hand tiene un valor mayor a 21, haga que la clase Hand verifique los Ases y reduzca el valor de puntos a 1.
   4. Haga al programa crear una Mano del crupier contra la cual pueda jugar el usuario. El usuario debe tratar de acercarse lo más posible a 21 sin pasarse para vencer al Crupier. Si el Crupier tiene 16 o más, éste deberá dejar de tomar cartas.

**public class** Main {

**public static void** main(String args[]){ Deck d = new Deck(); d.print();

}

}

**public class** Deck {

Card[] cardArray = new Card[52];

Deck(){ //constructor

**int** suits = 4;

**int** cardType = 13;

**int** cardCount = 0;

**for**(**int** i = 1; i <= suits; i++)

**for**(**int** j = 1; j <= cardType; j++){ cardArray[cardCount] = new Card(i,j); cardCount++;

}

}

**public void** print(){

**for**(**int** i = 0; i < cardArray.length; i++) System.*out*.println(cardArray[i]);

}

}

**public class** Card{

String suit,name;

**int** points; Card(**int** n1, **int** n2){ suit = getSuit(n1); name = getName(n2);

points = getPoints(name);

}

**public** String toString(){

**return** "The " + name + " of " + suit;

}

**public** String getName(**int** i){

**if**(i == 1) **return** "Ace"; **if**(i == 2) **return** "Two"; **if**(i == 3) **return** "Three"; **if**(i == 4) **return** "Four"; **if**(i == 5) **return** "Five"; **if**(i == 6) **return** "Six";

**if**(i == 7) **return** "Seven"; **if**(i == 8) **return** "Eight"; **if**(i == 9) **return** "Nine"; **if**(i == 10) **return** "Ten"; **if**(i == 11) **return** "Jack";

**if**(i == 12) **return** "Queen"; **if**(i == 13) **return** "King"; **return** "error";

}

**public int** getPoints(String n){

**if**(n == "Jack" ||n == "Queen" ||n == "King"||n == "Ten")

**return** 10; **if**(n == "Two") **return** 2;

**if**(n == "Three")

**return** 3;

**if**(n == "Four")

**return** 4;

**if**(n == "Five")

**return** 5;

**if**(n == "Six")

**return** 6;

**if**(n == "Seven")

**return** 7;

**if**(n == "Eight")

**return** 8;

**if**(n == "Nine")

**return** 9; **if**(n == "Ace") **return** 1;

**return** -1;

}

**public** String getSuit(**int** i){ **if**(i == 1) **return** "Diamonds"; **if**(i == 2) **return** "Clubs";

**if**(i == 3) **return** "Spades"; **if**(i == 4) **return** "Hearts"; **return** "error";

}

}