

CIFP César Manrique.

Programación 1º de Desarrollo de Aplicaciones Web

Profesor: José David Díaz Díaz

Actividades de la Unidad 5: Tablas

Juan Carlos Francisco Mesa



Esta obra está licenciada bajo la Licencia Creative Commons Atribución 4.0 Internacional.
Para ver una copia de esta licencia, visite <http://creativecommons.org/licenses/by/4.0/> o
envíe una carta a Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Índice

Actividades.....	1
Desarrollo.....	2
Actividades de comprobación.....	2
5.1. Una tabla puede almacenar datos de distintos tipos, como por ejemplo enteros, booleanos, reales, etcétera :.....	2
5.2. En Java, la numeración de los índices que determina la identificación de cada elemento de una tabla comienza en :.....	2
5.3. Si en una tabla de 10 elementos utilizamos el elemento con índice 11 (que se encuentra fuera de rango):.....	2
5.4. ¿Qué método de la clase Arrays permite realizar una búsqueda dicotómica en una tabla?.3	3
5.5. Con respecto a las tablas, el operador new :.....	3
5.6. La forma de invocar al recolector de basura es:.....	3
5.7. La forma de conocer la longitud de una tabla t es mediante:.....	3
5.8. La comparación del contenido (los elementos) de dos tablas se realiza utilizando:.....	4
5.9. ¿Qué condición tiene que cumplir una tabla para que podamos realizar búsquedas dicotómicas en ella?.....	4
5.10. ¿Cuál es la principal diferencia entre Arrays.copyOf() y System.arraycopy()?.....	4
Actividades de Aplicación.....	5
5.11. Realiza la función: int[] buscarTodos (int t[], int clave), que crea y devuelve una tabla con todos los índices de los elementos donde se encuentra la clave de búsqueda.....	5
5.12. Escribe la función void desordenar (int t[]), que cambia de forma aleatoria los elementos contenidos en la tabla t.....	7
5.13. Modifica la Actividad de aplicación 5.12 para que la función no modifique la tabla	

que se pasa como parámetro y, en su lugar, cree y devuelva una copia de la tabla donde se han desordenado los valores de los elementos.....	9
5.14. El ayuntamiento de tu localidad te ha encargado una aplicación que ayude a realizar encuestas estadísticas para conocer el nivel adquisitivo de los habitantes del municipio.....	11
5.15. Debes desarrollar una aplicación que ayude a gestionar las notas de un centro educativo.....	13
5.18. Escribe un programa que solicite los elementos de una matriz de tamaño 4 x 4. La aplicación debe decidir si la matriz introducida corresponde a una matriz mágica, que es aquella donde la suma de los elementos de cualquier fila o de cualquier columna valen lo mismo.....	15

Actividades

Actividades de la Unidad 5: Tablas.

En este documento se detallan las actividades a realizar. Se entregará al profesor en la plataforma digital dos ficheros. Un primer fichero pdf con todas las actividades a realizar, el nombre del fichero será “unidad2 + nombre del alumno.pdf”. Añadir en el fichero pdf por cada actividad de programación dos capturas de pantalla, una del código y otra de su ejecución. También en el fichero pdf copiar todas las preguntas y las respuestas correctas de las actividades de comprobación. Además, entregar un segundo fichero comprimido con todos los códigos fuentes de cada actividad de programación realizada.

Todas las actividades resueltas se deberán de analizar y no se entregarán.

A continuación, detallamos las actividades a realizar:

- **Actividades propuestas.** No realizar ninguna.
- **Actividades de comprobación.** Realizarlas todas. Copiar todas las preguntas y sus respuestas correctas.
- **Actividades de aplicación.** Realizar las siguientes **5.11, 5.12, 5.13, 5.14, 5.15 y 5.18**.
- **Actividades de ampliación.** No realizar ninguna.

[Volver al índice](#)

Desarrollo

Actividades de comprobación.

5.1. Una tabla puede almacenar datos de distintos tipos, como por ejemplo enteros, booleanos, reales, etcétera :

- a) Cierto, las tablas siempre pueden almacenar datos de distintos tipos.
- b) Falso, las tablas solo pueden almacenar datos de un único tipo.**
- c) Puede almacenar datos de distintos tipos siempre que estos sean numéricos.
- d) Puede almacenar datos de distintos tipos siempre que la longitud de los datos sea idéntica.

[Volver al índice](#)

5.2. En Java, la numeración de los índices que determina la identificación de cada elemento de una tabla comienza en :

- a) Cero.**
- b) Uno.
- c) Depende del tipo de dato de la tabla.
- d) Es configurable por el usuario .

[Volver al índice](#)

5.3. Si en una tabla de 10 elementos utilizamos el elemento con índice 11 (que se encuentra fuera de rango):

- a) Al salir del rango de la longitud, Java redimensiona la tabla de forma automática.
- b) No es posible y produce un error.**
- c) Las tablas tienen un comportamiento circular y utilizar el índice 11 es idéntico a utilizar el índice 1.
- d) Ninguna de las anteriores respuestas es cierta .

[Volver al índice](#)

5.4. ¿Qué método de la clase Arrays permite realizar una búsqueda dicotómica en una tabla?

- a) Arrays.search().
- b) Arrays.find().
- c) **Arrays.binarySearch()**.
- d) Cualquiera de los métodos anteriores realiza una búsqueda.

[Volver al índice](#)

5.5. Con respecto a las tablas, el operador new :

- a) Destruye, crea y redimensiona tablas.
- b) Destruye y crea tablas.
- c) **Crea tablas.**
- d) Destruye las tablas .

[Volver al índice](#)

5.6. La forma de invocar al recolector de basura es:

- a) Mediante System.garbageCollector().
- b) Mediante el operador new.
- c) Mediante Arrays.garbageCollector().
- d) **Ninguna de las anteriores respuestas es correcta.**

[Volver al índice](#)

5.7. La forma de conocer la longitud de una tabla t es mediante:

- a) t.size.
- b) t.elements.
- c) **t.length.**
- d) Arrays.size (t) .

[Volver al índice](#)

5.8. La comparación del contenido (los elementos) de dos tablas se realiza utilizando:

a) Arrays.compare().

b) El operador ==.

c) **Arrays.equals().**

d) Arrays.same() .

[Volver al índice](#)

5.9. ¿Qué condición tiene que cumplir una tabla para que podamos realizar búsquedas dicotómicas en ella?

a) Que esté ordenada.

b) Que esté ordenada y sea una tabla de enteros.

c) Que no esté ordenada.

d) No importa si la tabla está ordenada, lo realmente importante es que sea de algún tipo numérico .

[Volver al índice](#)

5.10. ¿Cuál es la principal diferencia entre Arrays.copyOf() y System.arraycopy()?

a) No existe diferencia alguna, ambos métodos son idénticos.

b) Arrays.copyOf() copia mientras System.arraycopy() copia y compara.

c) Arrays.copyOf() copia entre tablas existentes mientras System.arraycopy() crea una nueva tabla y copia en ella.

d) **Arrays.copyOf() crea una nueva tabla y copia en ella mientras System.arraycopy() solo copia entre tablas ya creadas .**

[Volver al índice](#)

Actividades de Aplicación.

5.11. Realiza la función: int[] buscarTodos (int t[], int clave), que crea y devuelve una tabla con todos los índices de los elementos donde se encuentra la clave de búsqueda.

En el caso de que clave no se encuentre en la tabla t, la función devolverá una tabla vacía.

```

import java.util.Scanner;
import java.util.Arrays;

/*
 * @author juancfm
 */
public class Main {

    /**
     * Realiza la función: int[ ] buscarTodos (int t[ ], int clave), que crea y
     * devuelve una tabla con todos los índices de los elementos donde se
     * encuentra la clave de búsqueda. En el caso de que clave no se encuentre
     * en la tabla t, la función devolverá una tabla vacía.
     */
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int clave;
        int[] t, result;
        t = new int[20];
        poblarArray(t);
        result = new int[0];
        System.out.print("Introduzca un número de 1 a 20 para ver cuantas "
                + "\'n\'veces aparece en nuestro array aleatorio de 20 números: ");
        clave = sc.nextInt();
        result = buscarTodos(t, clave);
        if (result.length == 1) {
            System.out.println("\nHay " + result.length + " coincidencia y se "
                    + "\'n\'encuentra en el indice: \n"
                    + Arrays.toString(result) + "\ndel array");
            System.out.println("(Recordar que los indices empiezan en 0)");
        }
    }
}

```



```

} else if (result.length > 1) {
    System.out.println("\nHay " + result.length + " coincidencias y se "
            + "\'n\'encuentran en los indices: \n"
            + Arrays.toString(result) + "\ndel array");
} else {
    System.out.println("\nNo hay coincidencias para este "
            + "número en el array " + Arrays.toString(result));
}

System.out.println("\nArray aleatorio:");
System.out.println(Arrays.toString(t));
}

private static int[] buscarTodos(int[] t, int clave) {
    int[] result = new int[0];
    for (int i = 0; i < t.length; i++) {
        if (t[i] == clave) {
            result = Arrays.copyOf(result, result.length + 1);
            result[result.length - 1] = i;
        }
    }
    return result;
}

private static void poblarArray(int[] t) {
    for (int i = 0; i < t.length; i++) {
        t[i] = (int) (Math.random() * 20) + 1;
    }
}
}

```



```

Output - Main (run) ×

run:
Introduzca un número de 1 a 20 para ver cuantas
veces aparece en nuestro array aleatorio de 20 números: 12

Hay 2 coincidencias y se
encuentran en los indices:
[8, 10]
del array

Array aleatorio:
[17, 18, 2, 15, 3, 17, 11, 8, 12, 5, 12, 9, 14, 11, 16, 8, 17, 13, 8, 16]
BUILD SUCCESSFUL (total time: 3 seconds)

```

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×
run:
Introduzca un número de 1 a 20 para ver cuantas
veces aparece en nuestro array aleatorio de 20 números: 12

Hay 1 coincidencia y se
encuentra en el índice:
[0]
del array
(Recordar que los índices empiezan en 0)

Array aleatorio:
[12, 3, 17, 2, 5, 5, 11, 10, 19, 5, 17, 8, 13, 1, 8, 10, 20, 4, 4, 14]
BUILD SUCCESSFUL (total time: 3 seconds)
```

```
Output - Main (run) ×
run:
Introduzca un número de 1 a 20 para ver cuantas
veces aparece en nuestro array aleatorio de 20 números: 12

No hay coincidencias para este número en el array []

Array aleatorio:
[15, 11, 15, 18, 7, 18, 19, 8, 16, 6, 13, 10, 15, 11, 9, 2, 8, 15, 18, 9]
BUILD SUCCESSFUL (total time: 16 seconds)
```

[Volver al índice](#)

Actividades de la Unidad 5: Tablas

5.12. Escribe la función void desordenar (int t[]), que cambia de forma aleatoria los elementos contenidos en la tabla t.

Si la tabla estuviera ordenada, dejaría de estarlo.

```
package main;

import java.util.Scanner;
import java.util.Arrays;
public class Main {

    /**
     * Escribe la función void desordenar (int t[ ]), que cambia de forma
     * aleatoria los elementos contenidos en la tabla t. Si la tabla estuviera
     * ordenada, dejaría de estarlo.
     */
    public static void main(String[] args) {
        Scanner sc = new Scanner(source.System.in);
        int[] myArray;
        String option = "";

        while (!option.equals(anObject:"a") && !option.equals(anObject:"b")) {
            System.out.println("Indique si desea:");
            System.out.println("a)Introducir su propio array.");
            System.out.println("b)Generar un array aleatorio con números "
                + "entre 1 y 50.");
            option = sc.nextLine();
        }

        System.out.print("\nIndique ahora la longitud del array: ");
        int num = sc.nextInt();
        myArray = new int[num];
    }
}
```

1

```
/*
 * Prepara el poblado manual del array por parte del usuario
 *
 * @param myArray
 */
private static void poblarManual(int[] myArray) {
    Scanner scl = new Scanner(source.System.in);
    for (int i = 0; i < myArray.length; i++) {
        System.out.print("\nIntroduzca un valor para el elemento con "
            + "indice " + i + ": ");
        myArray[i] = scl.nextInt();
    }
    // return myArray;
}

/*
 * Prepara el poblado aleatorio del array con números enteros entre 1 y 50
 *
 * @param myArray
 */
private static void poblarArray(int[] myArray) {
    for (int i = 0; i < myArray.length; i++) {
        myArray[i] = (int) (Math.random() * 50) + 1;
    }
}
```

3

```
/*
 * Se determina cual método se usa para poblar array
 */
if (option.equals(anObject:"a")) {
    poblarManual(myArray);
    System.out.println("\nArray generado por el usuario: ");
} else {
    System.out.println("\nArray generado aleatoriamente: ");
    poblarArray(myArray);
}

// Se imprime el array resultante
System.out.println(Arrays.toString(as:myArray));

/*
 * Permite al usuario desordenar el array tantas veces como pulse la
 * tecla "d". "t" para parar la ejecución
 */
while (!option.equals(anObject:"t")) {
    System.out.print("Pulse (d)esordenar o (t)erminar: ");
    option = sc.nextLine();
    if (option.equals(anObject:"d")) {
        desordenar(myArray);
    }
}

sc.close();
System.exit(status:0);
}
```

2

```
/*
 * Desordena el array que se pasa como parámetro y lo muestra desordenado.
 * Para cada elemento i del array se generó un número aleatorio j, que está
 * dentro del rango de índices del array, procediendo luego a intercambiar
 * el elemento i por el elemento j, lo que produce una mezcla aleatoria de
 * los elementos
 *
 * @param myArray
 */
private static void desordenar(int[] myArray) {
    int j, comodin;
    for (int i = 0; i < myArray.length; i++) {
        j = (int) (Math.random() * myArray.length + 1) - 1;
        comodin = myArray[i];
        myArray[i] = myArray[j];
        myArray[j] = comodin;
    }
    System.out.println("\n" + Arrays.toString(as:myArray));
}
```

4

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×
run:
Indique si desea:
    a      Introducir su propio array.
    b      Generar un array aleatorio con números entre 1 y 50.
a

Indique ahora la longitud del array: 5

Introduzca un valor para el elemento con indice 0: 4

Introduzca un valor para el elemento con indice 1: 5

Introduzca un valor para el elemento con indice 2: 6

Introduzca un valor para el elemento con indice 3: 9

Introduzca un valor para el elemento con indice 4: 8

Array generado por el usuario:
[4, 5, 6, 9, 8]
Pulse (d)esordenar o (t)erminar: d

[6, 9, 4, 5, 8]
Pulse (d)esordenar o (t)erminar: d

[8, 9, 6, 5, 4]
Pulse (d)esordenar o (t)erminar: t
BUILD SUCCESSFUL (total time: 42 seconds)
```

[Volver al índice](#)

Actividades de la Unidad 5: Tablas

5.13. Modifica la Actividad de aplicación 5.12 para que la función no modifique la tabla que se pasa como parámetro y, en su lugar, cree y devuelva una copia de la tabla donde se han desordenado los valores de los elementos.

```

import java.util.Scanner;
import java.util.Arrays;
public class Main {
    /**
     * Escribe la función void desordenar (int t[]), que cambia de forma
     * aleatoria los elementos contenidos en la tabla t.
     * Si la tabla estuviera ordenada, dejaría de estarlo.
     */
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] myArray;
        String option = "";
        while (!option.equals("a") && !option.equals("b")) {
            System.out.println("Indique si desea:");
            System.out.println("a)Introducir su propio array.");
            System.out.println("b)Generar un array aleatorio con números "
                + "entre 1 y 50.");
            option = sc.nextLine();
        }

        System.out.print("\nIndique ahora la longitud del array: ");
        int num = sc.nextInt();
        myArray = new int[num];
    }
}

```

1

```

/**
 * Prepara el poblado manual del array por parte del usuario
 *
 * @param myArray
 */
private static void poblarManual(int[] myArray) {
    Scanner sc1 = new Scanner(System.in);
    for (int i = 0; i < myArray.length; i++) {
        System.out.print("Introduce un valor para el elemento con "
            + "indice " + i + ": ");
        myArray[i] = sc1.nextInt();
    }
    // return myArray;
}

/**
 * Prepara el poblado aleatorio del array con números enteros entre 1 y 50
 *
 * @param myArray
 */
private static void poblarArray(int[] myArray) {
    for (int i = 0; i < myArray.length; i++) {
        myArray[i] = (int) (Math.random() * 50) + 1;
    }
}

```

3

```

    /**
     * Se determina cual método se usa para poblar array
     */

    if (option.equals("a")) {
        poblarManual(myArray);
        System.out.println("\nArray generado por el usuario: ");
    } else {
        System.out.println("\nArray generado aleatoriamente: ");
        poblarArray(myArray);
    }

    // Se imprime el array resultante
    System.out.println(Arrays.toString(myArray));

    /**
     * Permite al usuario desordenar el array tantas veces como pulse la
     * tecla "d". "t" para parar la ejecución
     */
    while (!option.equals("t")) {
        System.out.print("Pulse (d)esordenar o (t)erminar: ");
        option = sc.nextLine();
        if (option.equals("d")) {
            desordenar(myArray);
        }
    }

    sc.close();
    System.exit(0);
}

```

2

```

    /**
     * Desordena el array que se pasa como parámetro y lo muestra desordenado.
     * Se creó una copia del array y luego para cada elemento i del array se
     * generó un número aleatorio j, que está dentro del rango de índices del
     * array, procediendo luego a intercambiar el elemento i por el elemento j,
     * lo que produce una mezcla aleatoria de los elementos
     *
     * @param myArray
     */
    private static void desordenar(int[] myArray) {
        int[] aux = new int[myArray.length];
        System.arraycopy(myArray, 0, aux, 0, myArray.length);
        int j, comodin;
        for (int i = 0; i < aux.length; i++) {
            j = (int) (Math.random() * myArray.length + 1) - 1;
            comodin = aux[i];
            aux[i] = aux[j];
            aux[j] = comodin;
        }
        System.out.println("\n" + Arrays.toString(aux));
    }
}

```

4

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×

▶ run:
Indique si desea:
    a      Introducir su propio array.
    b      Generar un array aleatorio con números entre 1 y 50.
▶ b

Indique ahora la longitud del array: 10

Array generado aleatoriamente:
[15, 7, 3, 47, 18, 9, 49, 26, 17, 19]
Pulse (d)esordenar o (t)erminar: d

[49, 19, 17, 9, 7, 26, 15, 3, 47, 18]
Pulse (d)esordenar o (t)erminar: d

[47, 9, 18, 3, 15, 7, 17, 19, 26, 49]
Pulse (d)esordenar o (t)erminar: t
BUILD SUCCESSFUL (total time: 40 seconds)
```

[Volver al índice](#)

Actividades de la Unidad 5: Tablas

5.14. El ayuntamiento de tu localidad te ha encargado una aplicación que ayude a realizar encuestas estadísticas para conocer el nivel adquisitivo de los habitantes del municipio.

Para ello, tendrás que preguntar el sueldo a cada persona encuestada. A priori, no conoces el número de encuestados. Para finalizar la entrada de datos, introduce un sueldo con valor -1. Una vez terminada la entrada de datos, muestra la siguiente información:

- Todos los sueldos introducidos ordenados de forma decreciente.
- El sueldo máximo y mínimo.
- La media de los sueldos.

```
package main;

import java.util.Scanner;
import java.util.Arrays;

public class Main {
    /**
     * El ayuntamiento de tu localidad te ha encargado una aplicación que ayude a realizar encuestas estadísticas para conocer el nivel adquisitivo de los habitantes del municipio.
     * Para ello, tendrás que preguntar el sueldo a cada persona encuestada. A priori, no conoces el número de encuestados. Para finalizar la entrada de datos, introduce un sueldo con valor -1. Una vez terminada la entrada de datos, muestra la siguiente información:
     * • Todos los sueldos introducidos ordenados de forma decreciente.
     * • El sueldo máximo y mínimo.
     * • La media de los sueldos.
     */
    public static void main(String[] args) {
        double[] sueldos = new double[0];
        sueldos = cargarDatos(sueldos);
        ordenarSueldos(sueldos);
        sueldosMaxMin(sueldos);
        mediasueldos(sueldos);
        System.exit( status:0 );
    }
}
```



```
/*
 * Ordena el array de forma decreciente.
 * El método sort los ordena de manera predeterminada de manera ascendente,
 * de modo que una vez ordenado de esa manera pasamos a invertir el array
 * intercambiando el primer valor con el último, el segundo con el penúltimo, el tercero con el antepenúltimo y así sucesivamente hasta llegar al centro del array.
 */
@param sueldos
private static void ordenarSueldos(double[] sueldos) {
    Arrays.sort(sueldos);
    double aux;
    for (int i = 0, j = sueldos.length - 1; i <= (int) ((sueldos.length - 1) / 2); i++, j--) {
        aux = sueldos[i];
        sueldos[i] = sueldos[j];
        sueldos[j] = aux;
    }
    System.out.println("\nLos sueldos en orden decreciente son: ");
    System.out.println(Arrays.toString(sueldos));
}
```



```
/*
 * Calcula la media de los sueldos y la imprime en pantalla
 *
 * @param sueldos
 */
private static void mediasueldos(double[] sueldos) {
    double sum = 0;
    for (double sueldo : sueldos) {
        sum += sueldo;
    }
    System.out.println("\nLa media de sueldos es: "
        + sum / (sueldos.length + 1));
}

/*
 * Calcula el máximo y mínimo sueldo presente en el array y los presenta en pantalla
 *
 * @param sueldos
 */
private static void sueldosMaxMin(double[] sueldos) {
    double max, min;
    max = min = sueldos[0];
    for (double sueldo : sueldos) {
        if (max < sueldo)
            max = sueldo;
        if (min > sueldo)
            min = sueldo;
    }
    System.out.println("\nEl sueldo máximo es: " + max);
    System.out.println("El sueldo mínimo es: " + min);
}
```



```
/*
 * Carga de datos que conforman el array.
 * @param sueldos
 * @return double[]
 */
private static double[] cargarDatos(double[] sueldos) {
    double sueldo = 0;
    Scanner sc = new Scanner( source: System.in );

    System.out.println("Vamos a introducir los sueldos de las personas encuestadas");
    System.out.println("Cuando deseé detener la carga de los datos introduce \"-1\"");
    do {
        System.out.print("\nIntroduzca el sueldo de la persona " + (sueldos.length + 1) + ": ");
        sueldo = sc.nextDouble();
        if (sueldo != -1) {
            sueldos = Arrays.copyOf( original: sueldos, sueldos.length + 1 );
            sueldos[sueldos.length - 1] = sueldo;
        }
    } while (sueldo != -1);
    System.out.println("\nLos sueldos en el orden original son: ");
    System.out.println(Arrays.toString(sueldos));

    return sueldos;
}
```



Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×
run:
Vamos a introducir los sueldos de las personas encuestadas
Cuando desee detener la carga de los datos introduzca "-1".

Introduzca el sueldo de la persona 1: 48,5
Introduzca el sueldo de la persona 2: 49,7
Introduzca el sueldo de la persona 3: 98,3
Introduzca el sueldo de la persona 4: 96
Introduzca el sueldo de la persona 5: 100
Introduzca el sueldo de la persona 6: -1

Los sueldos en el orden original son:
[48.5, 49.7, 98.3, 96.0, 100.0]

Los sueldos en orden decreciente son:
[100.0, 98.3, 96.0, 49.7, 48.5]

El sueldo máximo es: 100.0
El sueldo mínimo es: 48.5
La media de sueldos es: 65.41666666666667
BUILD SUCCESSFUL (total time: 25 seconds)
```

[Volver al índice](#)

Actividades de la Unidad 5: Tablas

5.15. Debes desarrollar una aplicación que ayude a gestionar las notas de un centro educativo.

Los alumnos se organizan en grupos compuestos por 5 personas. Leer las notas (números enteros) del primer, segundo y tercer trimestre de un grupo. Debes mostrar al final la nota media del grupo en cada trimestre y la media del alumno que se encuentra en una posición dada (que el usuario introduce por teclado).

```

package main;

import java.util.Scanner;
import java.util.Arrays;

public class Main {
    public static void main(String[] args) {
        /*
         * 5.15. Debes desarrollar una aplicación que ayude a gestionar las
         * notas de un centro educativo.
         * Los alumnos se organizan en grupos compuestos por 5 personas. Leer
         * las notas (números enteros) del primer, segundo y tercer trimestre
         * de un grupo. Debes mostrar al final la nota media del grupo en cada
         * trimestre y la media del alumno que se encuentra en una posición
         * dada (que el usuario introduce por teclado).
        */

        boolean continuar = true;
        int persona;
        Scanner sc = new Scanner(System.in);

        int[][] notas = {
            { 1, 8, 9 }, // Notas Persona 1, posición 0
            { 9, 8, 4 }, // Notas Persona 2, posición 1
            { 10, 4, 9 }, // Notas Persona 3, posición 2
            { 7, 9, 8 }, // Notas Persona 4, posición 3
            { 10, 10, 10 } // Notas Persona 5, posición 4
        };
    }
}

```



```

/*
 * Le pregunta al usuario si desea cambiar las notas predeterminadas por
 * unas aleatorias, en caso afirmativo, se recorre el array bidimensional y
 * se asignan valores aleatorios enteros entre 1 y 10, y devuelve el array
 * con los nuevos datos.
 */
* @param notas
* @return int[][]
*/
private static int[][] cambioNotas(int[][] notas) {
    String option;
    System.out.println("Las notas actuales son: ");
    System.out.println(Arrays.deepToString(notas));
    System.out.println("Desea sustituir las notas con otras aleatorias? (s/n)");
    Scanner scl = new Scanner(System.in);
    option = scl.nextLine();
    if (option.equals("s")) {
        for (int i = 0; i < notas.length; i++) {
            for (int j = 0; j < notas[i].length; j++) {
                notas[i][j] = (int) (Math.random() * 10 + 1);
            }
        }
        System.out.println("Las nuevas notas son: ");
        System.out.println(Arrays.deepToString(notas) + "\n");
    }
    return notas;
}

```



```

notas = cambioNotas(notas); // Comprueba si se desean notas aleatorias
notasTrimestre(notas); // Muestra la media de notas trimestral del grupo

// Se mantiene en un bucle para mostrar las notas de un alumno en una
// posición que el usuario escoga, hasta que se introduzca un número
// menor a cero o mayor a cuatro
while (continuar) {
    System.out.print(
        "Para ver la nota de algún alumno pulse su posición (0-4) "
        + "\n(Introduzca otro número para salir): ");
    persona = sc.nextInt();
    if (persona > 0 && persona <= 4) {
        mostrarNotas(notas, persona); // Muestra las notas y el promedio de una persona
    } else {
        continuar = false;
    }
}

sc.close();
System.exit(0);
}

```



```

/*
 * Imprime por consola el promedio de notas agrupadas por trimestre
 */
* @param notas
*/
private static void notasTrimestre(int[][] notas) {
    double[] trim = new double[3];

    for (int i = 0; i < notas.length; i++) {
        for (int j = 0; j < notas[i].length; j++) {
            trim[j] += notas[i][j];
        }
    }

    System.out.println("El promedio de notas por trimestre es:");
    System.out.println("Primer Trimestre:" + trim[0] / notas.length);
    System.out.println("Segundo Trimestre:" + trim[1] / notas.length);
    System.out.println("Tercer Trimestre:" + trim[2] / notas.length + "\n");
}

/*
 * Muestra las notas de una persona determinada y su promedio de notas
 */
* @param notas
* @param persona
*/
private static void mostrarNotas(int[][] notas, int persona) {
    double prom = 0;

    for (int i = 0; i < notas[persona].length; i++) {
        prom += notas[persona][i];
    }
    System.out.print(
        "\nLas notas de la persona en la posición " + persona
        + " son: " + Arrays.toString(notas[persona]));
    System.out.println("\nSu promedio es: " + prom / 3 + "\n");
}

```



Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×
run:
Las notas actuales son:
[[1, 8, 9], [9, 8, 4], [10, 4, 9], [7, 9, 8], [10, 10, 10]]
¿Desea sustituir las notas con otras aleatorias?(s/n)
s
Las nuevas notas son:
[[3, 6, 1], [1, 4, 2], [9, 10, 2], [1, 4, 1], [9, 1, 10]]

El promedio de notas por trimestre es:
Primer Trimestre:      4.6
Segundo Trimestre:     5.0
Tercer Trimestre:       3.2

Para ver la nota de algún alumno pulse su posición (0-4)
(Introduzca otro número para salir): 2

Las notas de la persona en la posición 2 son: [9, 10, 2]
Su promedio es: 7.0

Para ver la nota de algún alumno pulse su posición (0-4)
(Introduzca otro número para salir): 9
BUILD SUCCESSFUL (total time: 47 seconds)
```

[Volver al índice](#)

Actividades de la Unidad 5: Tablas

5.18. Escribe un programa que solicite los elementos de una matriz de tamaño 4 x 4. La aplicación debe decidir si la matriz introducida corresponde a una matriz mágica, que es aquella donde la suma de los elementos de cualquier fila o de cualquier columna valen lo mismo.

```

package main;

import java.util.Scanner;
import java.util.Arrays;

public class Main {
    public static void main(String[] args) {
        /**
         * 5.18. Escribe un programa que solicite los elementos de una matriz
         * de tamaño 4 x 4. La aplicación debe decidir si la matriz introducida
         * corresponde a una matriz mágica, que es aquella donde la suma de los
         * elementos de cualquier fila o de cualquier columna valen lo mismo.
         */
        int[][] matriz = new int[4][4];
        Scanner sc = new Scanner(System.in);
        String option;
        boolean continuar = true;
    }
}

```

1

```

/**
 * Método para la introducción de los números que forman la matriz
 * @param matriz
 * @param sc
 */
private static void introducirNumeros(int[][] matriz, Scanner sc) {
    for (int i = 0; i < matriz.length; i++) {
        System.out.print(":" + i + "\n");
        for (int j = 0; j < matriz[i].length; j++) {
            System.out.print("\nVamos a introducir el valor para la fila " +
                            " " + i + ", columna " + j + ": ");
            matriz[i][j] = sc.nextInt();
        }
    }
    System.out.println("\nLa matriz introducida es :");
    mostrarMatriz(matriz);
}

/**
 * Muestra la matriz una vez introducidos los datos
 * @param matriz
 */
private static void mostrarMatriz(int[][] matriz) {
    for (int i = 0; i < matriz.length; i++) {
        System.out.print(":" + i + "\n");
        for (int j = 0; j < matriz[i].length; j++) {
            System.out.print("\t" + matriz[i][j]);
        }
    }
}

```

3

```

System.out.println(
    "\nVamos a solicitar 16 elementos para formar una "
    + "\nmatriz 4x4 y vamos a comprobar si es una matriz "
    + "mágica, es decir, que "
    + "\nla suma de cualquiera de sus filas o de sus "
    + "columnas es igual, una informa fácil de comprobarlo "
    + "sería introduciendo los 16 números iguales."
    + "\nEmpecemos: ");

// Creamos un bucle para que el usuario pueda probar varias matrices
while (continuar) {
    introducirNumeros(matriz, sc); // Método para introducir los datos
    comprobarMatriz(matriz); // Método que comprueba las matrices

    System.out.print("Desea comprobar otra matriz (s/n): ");
    option = sc.next();
    if (option.equals("n")) {
        System.out.println("\nDe acuerdo. Empecemos:");
    } else {
        continuar = false;
    }
}

sc.close();
System.exit(0);
}

```

2

```

/**
 * Comprueba la matriz para ver si es una matriz mágica
 * @param matriz
 */
private static void comprobarMatriz(int[][] matriz) {
    int[] columnas = new int[4];
    int[] filas = new int[4];
    for (int i = 0; i < matriz.length; i++) {
        System.out.print(":" + i + "\n");
        for (int j = 0; j < matriz[i].length; j++) {
            filas[i] += matriz[i][j];
            columnas[j] += matriz[i][j];
        }
    }
    System.out.println(
        Arrays.equals(filas, columnas) ?
            "Excelente, esa es una tabla mágica" :
            "Lo siento, sigue intentando.");
}

```

4

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×
run:
Vamos a solicitar 16 elementos para formar una
matriz 4x4 y vamos a comprobar si es una matriz mágica, es decir, que
la suma de cualquiera de sus filas o de sus columnas es igual, una
forma fácil de comprobarlo sería introduciendo los 16 números iguales.
Empecemos:

Vamos a introducir el valor para la fila 0, columna 0: 4
Vamos a introducir el valor para la fila 0, columna 1: 6
Vamos a introducir el valor para la fila 0, columna 2: 9
Vamos a introducir el valor para la fila 0, columna 3: 2

Vamos a introducir el valor para la fila 1, columna 0: 3
Vamos a introducir el valor para la fila 1, columna 1: 5
Vamos a introducir el valor para la fila 1, columna 2: 4
Vamos a introducir el valor para la fila 1, columna 3: 9
```

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×
▶ Vamos a introducir el valor para la fila 2, columna 0: 6
▶ Vamos a introducir el valor para la fila 2, columna 1: 8
▶ Vamos a introducir el valor para la fila 2, columna 2: 2
▶ Vamos a introducir el valor para la fila 2, columna 3: 3

Vamos a introducir el valor para la fila 3, columna 0: 4
Vamos a introducir el valor para la fila 3, columna 1: 5
Vamos a introducir el valor para la fila 3, columna 2: 6
Vamos a introducir el valor para la fila 3, columna 3: 8

La matriz introducida es :

    4      6      9      2
    3      5      4      9
    6      8      2      3
    4      5      6      8

Lo siento, siga intentando.
Desea comprobar otra matriz (s/n): s

De acuerdo. Empecemos:
```

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×

▶▶▶▶▶ Vamos a introducir el valor para la fila 0, columna 0: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 0, columna 1: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 0, columna 2: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 0, columna 3: 5

▶▶▶▶▶ Vamos a introducir el valor para la fila 1, columna 0: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 1, columna 1: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 1, columna 2: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 1, columna 3: 5

▶▶▶▶▶ Vamos a introducir el valor para la fila 2, columna 0: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 2, columna 1: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 2, columna 2: 5
▶▶▶▶▶ Vamos a introducir el valor para la fila 2, columna 3: 5
```

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×
▶ Vamos a introducir el valor para la fila 3, columna 0: 5
▶ Vamos a introducir el valor para la fila 3, columna 1: 5
▶ Vamos a introducir el valor para la fila 3, columna 2: 5
▶ Vamos a introducir el valor para la fila 3, columna 3: 5
La matriz introducida es :
      5      5      5      5
      5      5      5      5
      5      5      5      5
      5      5      5      5

Excelente, esa es una tabla mágica
Desea comprobar otra matriz (s/n): s

De acuerdo. Empecemos:

Vamos a introducir el valor para la fila 0, columna 0: 4
Vamos a introducir el valor para la fila 0, columna 1: 5
Vamos a introducir el valor para la fila 0, columna 2: 6
Vamos a introducir el valor para la fila 0, columna 3: 7
```

Actividades de la Unidad 5: Tablas

```
Output - Main (run) ×

▶ Vamos a introducir el valor para la fila 1, columna 0: 5
▶ Vamos a introducir el valor para la fila 1, columna 1: 6
▶ Vamos a introducir el valor para la fila 1, columna 2: 7
▶ Vamos a introducir el valor para la fila 1, columna 3: 4

Vamos a introducir el valor para la fila 2, columna 0: 6
Vamos a introducir el valor para la fila 2, columna 1: 7
Vamos a introducir el valor para la fila 2, columna 2: 4
Vamos a introducir el valor para la fila 2, columna 3: 5

Vamos a introducir el valor para la fila 3, columna 0: 7
Vamos a introducir el valor para la fila 3, columna 1: 4
Vamos a introducir el valor para la fila 3, columna 2: 5
Vamos a introducir el valor para la fila 3, columna 3: 6
```

```
Output - Main (run) ×

▶ La matriz introducida es :
▶
    4      5      6      7
    5      6      7      4
    6      7      4      5
    7      4      5      6

Excelente, esa es una tabla mágica
Desea comprobar otra matriz (s/n): n
BUILD SUCCESSFUL (total time: 1 minute 37 seconds)
```

[Volver al índice](#)