**GUÍA DE TRABAJO**

**Arreglos Bidimensionales o Matrices**

Asignatura : Introducción a la Programación. Secciones 1, 2 y 3. Semestre: Otoño 2013

Profesores : Marcela Pinto F. – Alexander Fell S. – Diego Sepúlveda B.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**PARTE I**

1. Indique el o los errores, si los hay en los siguientes códigos de programa Java y realice la corrección correspondiente:
   1. Suponga que int[] b = new int[10][5];

for (int(i=0; i<= b.length; i++)

for (int(j=0; j<= b[i].length; j++)

b[i][j] = 1;

* 1. Suponga que int[][] a = {{1,2},{3,4}};

a[2,2] = 5;

1. Para el siguiente código de programa, dibuje la matriz ventas de 4x4 y vaya llenando de ceros la matriz, de acuerdo a cómo se hace la asignación en el código (si hubiesen errores, indíquelos y corríjalos):

int[][] ventas = new int[4][4];

for (int i=0; i<ventas[0].length; i++)

for (int j=0; j<ventas.length; j++) {

ventas[j,i] = 0;

}

En los 3 ejercicios que siguen indique claramente cómo se recorren las matrices definidas, cómo se generan las matrices resultantes y lo que se despliega en pantalla.

1. Generación de matriz transpuesta.

**public** **class** PruebaArreglosBidimensionales {

**public** **static** **void** main(String[] args) {

**int**[][] mat = {{10,12,33,14,25},{20,30,40,50,60},{34,45,56,67,78},{4,15,16,70,28}};

System.*out*.println("\nLa matriz original es: ");

**for** (**int** i=0; i<mat.length; i++) {

**for** (**int** j=0; j<mat[0].length; j++) {

System.*out*.print(mat[i][j] + " ");

}

System.*out*.println();

}

System.*out*.println();

System.*out*.println("\n\nLa matriz transpuesta es: ");

**for** (**int** j=0; j<mat[0].length; j++) {

**for** (**int** i=0; i<mat.length; i++) {

System.*out*.print(mat[i][j] + " ");

}

System.*out*.println();

}

System.*out*.println();

}

}

1. Suma de matrices.

**public** **class** PruebaArreglosBidimensionales {

**public** **static** **void** main(String[] args) {

**int**[][] matA = {{1,2,3,4},{2,3,4,5},{3,4,5,6},{4,5,6,7}};

**int**[][] matB = {{10,19,88,15},{29,68,22,35},{18,23,16,12},{26,40,50,44}};

**int**[][] matC;

matC = **new** **int**[matA.length][matA.length];

**for** (**int** i=0; i<matA.length; i++) {

**for** (**int** j=0; j<matA.length; j++) {

matC[i][j] = matA[i][j] + matB[i][j];

}

}

System.*out*.println("\nLa matriz resultante de la suma es: ");

**for** (**int** i=0; i<matC.length; i++) {

**for** (**int** j=0; j<matC[0].length; j++) {

System.*out*.print(matC[i][j] + " ");

}

System.*out*.println();

}

System.*out*.println();

}

}

1. Multiplicación de matrices.

**public** **class** PruebaArreglosBidimensionales {

**public** **static** **void** main(String[] args) {

**int**[][] matA = {{1,2,3,4},{2,3,4,5},{3,4,5,6}};

**int**[][] matB = {{8,7,6,5,4},{7,6,5,4,3},{6,5,4,3,2},{5,4,3,2,1,0}};

**int**[][] matC;

matC = **new** **int**[matA.length][matB[0].length];

**for** (**int** i=0; i<matA.length; i++) {

**for** (**int** j=0; j<matB[0].length; j++) {

**for** (**int** k=0; k<matA[0].length; k++) {

matC[i][j] += matA[i][k] \* matB[k][j];

}

}

}

System.*out*.println("\nLa matriz resultante de la multiplicacion es: ");

**for** (**int** i=0; i<matC.length; i++) {

**for** (**int** j=0; j<matC[0].length; j++) {

System.*out*.print(matC[i][j] + " ");

}

System.*out*.println();

}

System.*out*.println();

}

}

**PARTE II**

Para cada uno de los ejercicios que siguen construya un programa Java que cumpla con lo solicitado.

1. Cree una matriz cuadrada de orden 7 (7x7) y asígnele 5 a todos y cada uno de sus componentes o elementos.
2. Al programa resultante del ejercicio 6) agregue el despliegue de los elementos de la tercera fila.
3. Al programa resultante del ejercicio 6) agregue el despliegue de los elementos de la primera columna.
4. Al programa resultante del ejercicio 6) agregue el despliegue de la matriz en pantalla de manera que tenga la siguiente apariencia:

MATRIZ GENERADA

5 5 5 5 5 5 5

5 5 5 5 5 5 5

5 5 5 5 5 5 5

5 5 5 5 5 5 5

5 5 5 5 5 5 5

5 5 5 5 5 5 5

5 5 5 5 5 5 5

1. Cree una matriz cuadrada de orden 5 y complétela con los números naturales partiendo desde 1, de manera que el elemento ubicado en [1,1] sea 1, el ubicado en [1,2] un 2, el ubicado en [1,3] un 3,…, el elemento ubicado en [2,1] un 6, etc. Despliegue la matriz como en el ejercicio 7).
2. Construya una matriz similar a la del ejercicio anterior, pero en este caso debe ser de 5 \* 6. Despliegue la suma de los números almacenados en la matriz.
3. Lea cadenas de caracteres (las cuales deberán ser ingresadas por el usuario) y almacénelas en una matriz de 3x4. Determine y despliegue el número de cadenas (strings) con 5 o más caracteres.
4. Cree una matriz de 10x10 y almacene números decimales leídos desde teclado. A continuación determine la posición [fila, columna] del mayor número almacenado. Suponga inicialmente que todos los números son diferentes. En una segunda versión de su programa considere la presencia de números repetidos, en este caso, si dos o más elementos poseen el valor mayor, entonces despliegue la posición de todos ellos.

**PARTE III**

Para cada uno de los ejercicios que siguen construya un programa Java que cumpla con lo solicitado.

1. Construya una matriz cuadrada de orden 7 cuyos valores enteros deben leerse desde teclado. Calcule:
   1. La suma de cada fila y almacene los valores en un arreglo unidimensional,
   2. La suma de cada columna y almacene los valores en un arreglo unidimensional.
2. Construya una matriz cuadrada de orden 20 cuyos valores decimales deben leerse desde teclado. Sume las columnas y despliegue el valor promedio de ellas.
3. Similar al anterior, pero debe desplegar el número de la columna (valor entre 1 y 20, ambos inclusive) que contenga la suma mayor y el de la columna con la suma menor. Suposición: todas las columnas suman diferentes cantidades.
4. Similar al anterior pero eliminando la suposición, por lo que debe desplegar el o los números de la o las columnas correspondientes.
5. Construya una matriz de 5x5 cuyos valores enteros deben leerse. A continuación almacene la diagonal principal en un arreglo unidimensional. Finalmente, despliegue el arreglo resultante.
6. Construya una matriz cuadrada de orden 10 que almacene unos en la diagonal secundaria, y cero en las demás posiciones.
7. Construya una matriz de 6x8 con los valores numéricos ingresados por el usuario. Luego almacene toda la matriz en un arreglo unidimensional. Finalmente, despliegue el arreglo unidimensional resultante.
8. Construya una matriz de nxm con los valores numéricos ingresados por el usuario (n y m deben leerse). Luego almacene las n sumas correspondientes a cada una de las filas y las m sumas de las columnas en un arreglo unidimensional. Finalmente, despliegue el arreglo unidimensional resultante.
9. Construya una matriz cuadrada de orden n con los valores numéricos ingresados por el usuario (n debe leerse), y despliegue cuántos de los elementos son ceros, cuántos son positivos y cuántos son negativos.
10. Al programa anterior agregue el cálculo y despliegue de la suma de los elementos de la diagonal principal. Lo desplegado debe ser semejante a lo presentado a continuación:

Suponiendo que la matriz fuera: 1 24 5 2 14

10 3 10 8 10

7 7 12 18 6

19 5 6 0 14

5 12 13 25 8

Se debería desplegar:

1 + 3 + 12 + 0 + 8 = 24

1. Construya una matriz cuadrada de orden n cuyos elementos numéricos deben leerse, luego calcule y despliegue el promedio de los valores almacenados en la triangular superior (derecha), sin incluir la diagonal principal.
2. Lea una secuencia de 50 números, todos ellos con un valor entre 1 y 15 (en la secuencia vendrán números repetidos); y luego genere una matriz de 15 x 2, en su primera columna irá el número y en la segunda la cantidad de veces que se repitió dentro de la secuencia leída. Finalmente, despliegue el contenido de la matriz de tal modo que resulte claro para el usuario el significado de este.

**PARTE IV**

Para cada uno de los ejercicios que siguen construya un programa Java que cumpla con lo solicitado.

1. Sea una matriz de 24x7, en la cual se almacenen las temperaturas medidas cada hora durante una semana (de lunes a domingo). Se necesita calcular los promedios de cada día (lunes, martes, …) y guardar dichos promedios en un arreglo unidimensional para luego desplegar el nombre del día y el promedio correspondiente. Asimismo se necesita determinar y desplegar la temperatura promedio de la semana a las 14:00 horas.
2. Sea una matriz de 50x3 con el código de producto en la primera columna, su precio unitario en la segunda y el total de unidades existentes en la tercera. Se necesita, al ingresar el código y cantidad vendida de un producto cualquiera, se indique el total a pagar por parte del cliente y se rebaje la cantidad correspondiente del total de unidades existentes. Se deben realizar las siguientes validaciones: existencia de un código de producto, cantidad de unidades suficiente para realizar una venta y otras que estime conveniente.
3. El dueño de una cadena de tiendas de artículos deportivos desea controlar sus ventas usando un software ad hoc. Los datos de entrada por cada venta son :
   * Número de la tienda (valor entre 1 y 50, ambos inclusive).
   * Número del deporte al cual corresponde el artículo (valor entre 1 y 20, ambos inclusive).
   * Costo del artículo (en $).

Se requiere desplegar, al final del día, lo siguiente:

* 1. Ventas totales de cada tienda.
  2. Ventas totales por cada uno de los deportes.
  3. Ventas totales de la cadena de tiendas.

1. Se necesita leer datos referentes a las horas trabajadas cada día de la semana por los empleados de la empresa "XYZ", y almacenar tales datos en una matriz. Por cada empleado se tienen los siguientes antecedentes: Código del empleado (entero en el rango [1..100]), Horas trabajadas en horario normal, Horas trabajadas en sobretiempo. La empresa cancela un valor de $12.350.- bruto por hora normal y $18.500.- por hora de sobretiempo. Se debe emitir un informe en el cual, por cada empleado, se despliegue: código, cantidad de horas normales trabajadas, $ por horas normales, cantidad de horas de sobretiempo trabajadas, $ por horas de sobretiempo, total a pagar. Finalmente, se necesita conocer el monto total a pagar por parte de la empresa producto de las remuneraciones de todos los empleados.
2. La empresa de arriendo de automóviles “RentalCars”, desea una solución automática para calcular el gasto mensual de bencina de cada uno de sus vehículos. A continuación se presentan los antecedentes con que se cuenta:
   1. Por cada vehículo se tiene:
      1. Identificación (entero en el rango[10..95]).
      2. Rendimiento (km por litro), valor entero (se debería validar que sea siempre > 0).

Con estos datos se pueden procesar los viajes realizados durante el mes.

* 1. Por cada viaje se cuenta con los siguientes datos:
     1. Identificación del vehículo.
     2. Distancia entre el punto de partida y el de destino km), valor entero (se debería validar que sea siempre > 0). Un viaje implica un recorrido ida y vuelta.
  2. Por cada viaje se necesita calcular el total de litros de bencina utilizados.
  3. Un mismo vehículo puede realizar varios viajes en un mes, por lo cual se precisa acumular los litros de bencina usados en el mes.
  4. Finalmente, por cada vehículo se debe indicar: Identificación y Total de litros de bencina utilizados en el mes.

1. Se tienen los resultados de la votación de concejales de una comuna. Los datos a leer consisten en: código del candidato, código del sector y total de votos (todos enteros). Se desea obtener el resultado de la votación, indicando el código del candidato ganador y la cantidad de votos obtenidos. Asimismo se necesita una lista de los votos obtenidos por cada uno de los otros candidatos, ordenada de mayor a menor, indicando código del candidato y cantidad de votos. Se inscribieron 10 candidatos y la comuna cuenta con 15 sectores.
2. Una ferretería cuenta con un inventario de 20 productos que comercializa diariamente. Por cada producto se tienen los siguientes datos: código (entero en el rango [1..20]), nombre, número de unidades existentes, precio. Se necesita procesar un conjunto de ventas, a medida que estas se van produciendo. Por cada venta se debe leer el código del producto y la cantidad vendida, luego, si existe el saldo suficiente para la venta, se deberá indicar el nombre del producto y el total a pagar, en caso contrario indicar la situación mediante un mensaje. Cada vez que sea posible realizar una venta, se deberá rebajar la cantidad vendida del inventario.
3. Se tiene una secuencia de datos (que concluye con un 0) correspondientes a las visitas realizadas a un museo a fin de determinar las preferencias del público. El museo cuenta con 8 pisos, cada uno de los cuales dispone de 15 salas de exhibición pictórica. Se necesita procesar la secuencia con los recorridos de cada visitante, esto es piso y sala. Luego, una vez leída dicha secuencia, se deberá determinar la(s) sala(s) más visitada(s) y la(s) sala(s) menos visitada(s), en cada caso se deberá desplegar piso y sala. A continuación un ejemplo:

Datos de entrada (piso, sala):

1, 2

1, 4

3, 5

2, 7

8, 14

1, 2

0

Datos de salida:

La sala más visitada es la sala 2 del piso 1, con 2 visitas

La sala menos visitada es la sala 1 del piso 1, con 0 visitas

La sala menos visitada es la sala 3 del piso 1, con 0 visitas

La sala menos visitada es la sala 5 del piso 1, con 0 visitas

Etc.

1. Una tienda necesita procesar las compras mensuales realizadas por 130 clientes. Por cada cliente se cuenta con los siguientes datos: código (entero en el rango [1..30]), nombre, total compras ($) al contado y total compras ($) al crédito.

La tienda desea que se emita un informe con el siguiente formato:

Ventas por Cliente

Nombre Código Total compras Total compras

Cliente Cliente al contado al crédito

xxxxxxxxxxxxxxxx 99999 999999999 999999999

xxxxxxxxxxxxxxxx 99999 999999999 999999999

xxxxxxxxxxxxxxxx 99999 999999999 999999999

xxxxxxxxxxxxxxxx 99999 999999999 999999999

xxxxxxxxxxxxxxxx 99999 999999999 999999999

El informe debe estar ordenado en forma descendente por el código del cliente.

1. La Universidad TIM necesita implementar un sistema de reserva de salas. La institución cuenta con 50 salas en las que se deben dictar 113 asignaturas durante cada semestre. Hacen uso de sus dependencias 5.500 estudiantes y 52 profesores.

Los usuarios realizan una reserva ingresando los siguientes datos: Número de la sala, hora y asignatura, donde Número de la sala es un número entre 1 y 50, hora es un número entre 8 y 22 y, finalmente, asignatura es un string que indica el nombre de la asignatura para la cual se hace la reserva de sala. Las reservas siempre se realizan para el día siguiente.

Se supondrá que todos los datos son correctos, esto es, que no se requiere validarlos. Sin embargo, se puede intentar reservar una sala ya ocupada, por lo que se deberá rechazar una reserva cuando la sala solicitada en la hora y fecha indicada ya haya sido reservada.

Se debe además presentar el siguiente listado:

RESERVAS DE SALAS

|  |  |  |
| --- | --- | --- |
| SALA | ASIGNATURA | HORA |
| 99 | XXXXXXXXXXXXXX | 99 |
| 99 | XXXXXXXXXXXXXX | 99 |
| ... | ... | ... |
| 99 | XXXXXXXXXXXXXX | 99 |

Sala más solicitada: 99

Sala menos solicitada: 99

Asignatura con más reservas: XXXXXXXXXXXXXX

Las reservas deben mostrarse ordenadas por sala.

**PARTE V**

1. El lenguaje Logo hizo famoso el concepto de los gráficos de tortuga. Imagine una tortuga mecánica que camina por todo el cuarto, bajo el control de un programa Java. La tortuga sostiene un lápiz en una de dos posiciones, arriba o abajo. Mientras el lápiz esté abajo, el animalito va trazando figuras a medida que se va moviendo, y mientras el lápiz está arriba, se mueve libremente sin trazar nada.

En este ejercicio se necesita simular la operación de la tortuga y crear un bloc de dibujo computarizado.

Utilice un arreglo bidimensional de 20x20 al que se le llamará piso, que se deberá inicializar con ceros. A continuación recupere los comandos de un arreglo que los contiene. Se debe llevar registro de la posición actual de la tortuga en todo momento (ubicación en el piso y orientación – ubicación de la cabeza: hacia arriba, hacia abajo, hacia la izquierda, hacia la derecha-), y si el lápiz se encuentra arriba o abajo. Suponga que la tortuga siempre empieza en la posición (0,0) del piso, con su pluma hacia arriba.

El conjunto de comandos que se considerará forma parte del lenguaje a simular es:

|  |  |
| --- | --- |
| **Comando** | **Significado** |
| 1 | Mover lápiz hacia arriba |
| 2 | Mover lápiz hacia abajo |
| 3 | Girar a la derecha |
| 4 | Girar a la izquierda |
| 5,n | Avanzar hacia adelante n espacios |
| 6 | Desplegar el piso (matriz de 20x20) |
| 9 | Fin de la secuencia de comandos (centinela) |

Como ejemplo se presenta a continuación una secuencia de comandos (programa Logo), considerando que la tortuga se encuentra en algún lugar cerca del centro del piso. Se debe dibujar un cuadrado de 12 por 12, el cual se desplegará en pantalla:

2,

5,12

3

5,12

3

5,12

3

5,12

1

6

9

Indicaciones: A medida que la tortuga se vaya desplazando con la pluma hacia abajo, asigne 1 a los elementos apropiados del arreglo bidimensional piso. Cuando se dé el comando 6 (desplegar el piso en pantalla), siempre que haya un 1 en el arreglo, despliegue un asterisco. Siempre que haya un cero despliegue un caaracter blanco.

Ejercicio 36) extraído de: Deitel & Deitel, 2012, Cómo Programar en Java, 9ª edición. Pearson.