Matrices

Programación I

Objetivos del tema

- Desarrollar aplicaciones con uso de matrices de dos dimensiones
- Resolver problemas aplicando diseño descendente

Matrices

- La matriz es una estructura de datos que se representa mediante una colección de datos de un mismo tipo. Por ejemplo, para el caso de **matrices de dos dimensiones (con las que vamos a trabajar)**, una matriz con 6 enteros puede ser:

1	3	5
2	4	4

- Un arreglo es una colección de datos que tiene una dimensión:

1	3	5

- Y las consideraciones realizadas para arreglos también son válidas para matrices:
  - Para definir una matriz se debe indicar: el tipo de todos sus datos, la cantidad de datos que va tener determinada por dos cantidades correspondientes a filas y columnas, y un nombre de variable que representa a toda la estructura.
  - Una vez definido el tamaño o la cantidad de datos de la matriz se considera que no cambia.
  - A cada dato de la matriz se puede acceder en forma directa o aleatoria a través del nombre y la posición en cada dimensión (fila y columna) de donde se ubica el dato.
  - El uso de la variable matriz junto con la posición (fila, columna) se considera que es una variable del mismo tipo de datos con que fue creada la matriz.

Declaración de una matriz

- La declaración de una matriz tiene la forma **tipo_dato [][] nombre_matriz;** por ejemplo:

```
int [][] matentero;
//matentero es una matriz de enteros
```

- En la declaración anterior matenteros no tiene datos. Para hacer el espacio donde estarán los datos se realiza **nombre_matriz = new tipo_dato[CANTIDAD FILAS] [CANTIDAD COLUMNAS];** por ejemplo:

```
matentero = new int [MAXFILAS][MAXCOLUMNAS];
//Suponiendo que MAXFILAS Y MAXCOLUMNAS son constantes enteras > 0, la
  anterior sentencia asigno a matenteros espacio para contener
  MAXFILAS*MAXCOLUMNAS enteros.
```

- La asignación de espacio se puede hacer de manera explícita (**no se va a utilizar**) a partir de valores **nombre_arreglo = {{valor1, valor2,…,valorN},..,{…}};** ejemplo:

```
matentero = { {1, 3, 5},    //en la fila 0 tiene 1, 3 y 5
              {2, 4, 4}};   //en la fila 1 tiene 2, 4 y 4
```

Acceso a la matriz

- Para acceder a cada espacio de la matriz se utilizan dos variables de posición, fila y columna, donde la primera va desde **0** a la **CANTIDAD FILAS-1** y la segunda va desde **0** a la **CANTIDAD COLUMNAS-1**. Por ejemplo:

```
fila = 2; //suponiendo que fila es un entero
columna = 3; //suponiendo que columna es un entero
matentero[fila][columna] = 5;
//se accede a la posición [fila][columna] de la matriz y se le asigna
//un 5
//si fila y columna no están dentro de sus posibles rangos dará ERROR
```

- El acceso a una posición fila y columna mediante valores no se recomienda, siempre se utilizan variables enteras dentro del espacio asignado a la matriz.

Pasaje de matrices como parámetro en Java

- **En el pasaje de parámetros de matrices ocurre lo mismo que con arreglos (tipos no primitivos)**. Cuando se utiliza una matriz como parámetro se pasa una copia que se lo asigna a la variable parámetro que figura en la declaración. El valor original y la copia (direcciones de memoria) pueden acceder a la misma colección de datos que incluye. Así, **un método puede modificar el contenido de la variable de tipo no primitivo que figura en la invocación.**

Métodos con matrices

- Java y otros lenguajes permiten resolver una fila de la matriz como si fuera un arreglo.
- Cada fila de la matriz es un arreglo del mismo tipo de la matriz y de tamaño MAXCOLUMNAS.
- Así, para una matriz definida como:

```
int[][] mat = new int[MAXFILAS][MAXCOLUMNAS];
//si fila es un entero que va entre 0 y MAXFILAS-1
```

mat[fila] es un arreglo de enteros al que se le pueden aplicar los métodos desarrollados sobre arreglos.

- En los recorridos por fila esto permite simplificar la resolución; en los recorridos por columna en Java no se puede acceder a una columna sin antes acceder a una fila.

Carga de matrices

- Para simplificar los ejercicios vamos a suponer que siempre nos dan matrices cargadas con datos. Para hacer pruebas de como funciona un código pueden usar un método que cargue desde consola. O podrán optar por cargas aleatorias.
- A continuación se dan ejemplos de carga de matriz de int desde teclado, carga de matrices de int, char, y double de forma aleatorio, e impresión de dichas matrices.

Ejemplo 1: carga manual, aleatoria e impresión de matrices de char, int y double

```
/*Metodos de carga e impresión para reutilizar
*/
import java.util.Random;
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_8_Ejemplo_1 {
 public static final int MAXFILA = 4;
 public static final int MAXCOLUMNA = 5;
 public static final int MAXVALOR = 10;
 public static final int MINVALOR = 1;
 public static void main(String[] args) {
     char [][] matchar;
     int [][] matint;
     double [][] matdouble;
     matchar = new char[MAXFILA][MAXCOLUMNA];
     matint = new int[MAXFILA][MAXCOLUMNA];
     matdouble = new double[MAXFILA][MAXCOLUMNA];
     cargar_matriz_int(matint);
     imprimir_matriz_int(matint);
     cargar_matriz_aleatorio_char(matchar);
     cargar_matriz_aleatorio_int(matint);
     cargar_matriz_aleatorio_double(matdouble);
     imprimir_matriz_char(matchar);
     imprimir_matriz_int(matint);
     imprimir_matriz_double(matdouble);
 }
}
```

Ejemplo 2: opción sin arreglos

```
/*Hacer un programa que dado una matriz de enteros de tamano 4*5 que se encuentra precargada, imprima por pantalla el
   promedio de cada una de sus filas.
*/
import java.util.Random;
public class Clase_8_Ejemplo_2 {
   public static final int MAXFILA = 4;
   public static final int MAXCOLUMNA = 5;
   public static final int MAXVALOR = 10;
   public static final int MINVALOR = 1;
   public static void main(String[] args) {
        int [][] matint;
        matint = new int[MAXFILA][MAXCOLUMNA];
        cargar_matriz_aleatorio_int(matint);
        imprimir_matriz_int(matint);
        imprimir_promedios_filas(matint);
   }
   public static void imprimir_promedios_filas (int[][] mat){
        for (int fila = 0 ; fila < MAXFILA; fila++){
            System.out.println("Promedio de la fila "+fila+" es "+promedio_fila(mat,fila));
        }
   }
   public static int promedio_fila (int[][] mat, int fila){
        int promedio;
        int suma = 0;
        for (int columna = 0 ; columna < MAXCOLUMNA; columna++) {
            suma+=mat[fila][columna];
        }
        promedio = suma/MAXCOLUMNA;
        return promedio;
   }
}
```

Ejemplo 2

¿Qué hace el programa Clase_8_Ejemplo_2, cuando se ejecuta el método imprimir_promedios_filas(int[][] mat)?

Cuando se ejecuta el método imprimir_promedios_filas(int[][] mat) lo primero que hace es comenzar con una iteración. La idea es obtener el promedio de cada fila e imprimir el resultado.

Para el for define una variable fila = 0;

Pregunta fila<=MAXFILA, si es verdadero, entonces intenta imprimir un mensaje pero antes debe obtener el resultado de llamar a promedio_fila(mat,fila)

Cuando llama a promedio_fila(mat,fila) define promedio y suma = 0, y comienza Con un for.

Pregunta columna<=MAXCOLUMNA, si es verdadero, acumula en suma lo que hay En mat[fila][columna]. Incrementa columna y
Pregunta columna<=MAXCOLUMNA, si es verdadero, acumula en suma lo que hay En mat[fila][columna]. Incrementa columna y
…
Cuando termina calcula promedio=suma/MAXCOLUMNA (veces que sumo).
Y retorna el promedio. Termina y vuelve a donde lo llamaron

Ese valor promedio completa lo que faltaba para imprimir, imprime e incrementa Fila.

Pregunta fila<=MAXFILA, si es verdadero, entonces intenta imprimir un mensaje pero antes debe obtener el resultado de llamar a promedio_fila(mat,fila)
…

Cuando termina de iterar por fila no hay más sentencias para ejecutar y sale.
Vuelve al main de donde se lo invocó y continua con las sentencias que siguen.

```
public static void imprimir_promedios_filas
  (int[][] mat){
  for (int fila = 0 ; fila < MAXFILA; fila++){
            System.out.println("Promedio de la fila
  "+fila+" es "+promedio_fila(mat,fila));
  }
}
public static int promedio_fila (int[][] mat, int
  fila){
  int promedio;
  int suma = 0;
  for (int columna = 0 ; columna < MAXCOLUMNA;
  columna++) {
            suma+=mat[fila][columna];
  }
  promedio = suma/MAXCOLUMNA;
  return promedio;
}
}
```

Ejemplo 3: opción con arreglos

```
/*Hacer un programa que dado una matriz de enteros de tamano 4*5 que se encuentra precargada, imprima por pantalla el promedio
   de cada una de sus filas.
*/
import java.util.Random;
public class Clase_8_Ejemplo_3 {
   public static final int MAXFILA = 4;
   public static final int MAXCOLUMNA = 5;
   public static final int MAXVALOR = 10;
   public static final int MINVALOR = 1;
   public static void main(String[] args) {
        int [][] matint;
        matint = new int[MAXFILA][MAXCOLUMNA];
        cargar_matriz_aleatorio_int(matint);
        imprimir_matriz_int(matint);
        imprimir_promedios_filas(matint);
   }
   public static void imprimir_promedios_filas (int[][] mat){
        for (int fila = 0 ; fila < MAXFILA; fila++){
            //mat[fila] es un arreglo de enteros de tamano MAXCOLUMNA
            //en la declaración de promedio_arreglo, la variable parametro arr puede acceder a los mismos valores que mat[fila]
            System.out.println("Promedio de la fila "+fila+" es "+promedio_arreglo(mat[fila]));
        }
   }
   public static int promedio_arreglo (int[] arr){
        int promedio;
        int suma = 0;
        for (int pos = 0 ; pos < MAXCOLUMNA; pos++) {
            suma+=arr[pos];
        }
        promedio = suma/MAXCOLUMNA;
        return promedio;
   }
}
```

Ejemplo 4: no se puede resolver con arreglos

```
/*Hacer un programa que dado una matriz de enteros de tamano 4*5 que se encuentra precargada, imprima por pantalla el promedio
   de cada una de sus columnas.
    0 1 2
0  |1|3|5|
1  |2|4|4|
*/
public class Clase_8_Ejemplo_4 {
   public static final int MAXFILA = 4;
   public static final int MAXCOLUMNA = 5;
   public static void main(String[] args) {
        int [][] matint;
        matint = new int [MAXFILA][MAXCOLUMNA];
        //cargar_matriz_aleatorio_int(matint);
        //imprimir_matriz_int(matint);
        imprimir_por_pantalla_promedios_matriz(matint);
   }
   public static void imprimir_por_pantalla_promedios_matriz(int [][] mat){
        int promedio;
        for (int columna = 0; columna < MAXCOLUMNA; columna++){
            promedio = obtener_promedio_columna(mat,columna);
            System.out.println("el promedio de la columna "+columna+" es "+promedio);
        }
   }
   public static int obtener_promedio_columna(int [][] mat, int columna){
    int promedio,suma;
        suma = 0;
        for (int fila = 0; fila < MAXFILA; fila++){
        suma+=mat[fila][columna];
        }
        promedio=suma/MAXFILA;
        return promedio;
   }
}
```

Ejemplo 5

```
/*Hacer un programa que dado una matriz de enteros de tamano 4*5 que se
   encuentra precargada, encuentre la posicion fila,columna de un
   numero entero ingresado por el usuario. Si existe, muestre esa
   posicion por pantalla, o indique que no existe.
 */
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_8_Ejemplo_5 {
   public static int MAXFILA = 4;
   public static int MAXCOLUMNA = 5;
   public static void main(String[] args) {
    int numero;
         int [][] matint;
         matint = new int [MAXFILA][MAXCOLUMNA];
         //cargar_matriz_aleatorio_int(matint);
         //imprimir_matriz_int(matint);
         BufferedReader entrada = new BufferedReader(new
   InputStreamReader(System.in));
    try{
         System.out.println("ingrese un numero entero: ");
          numero = Integer.valueOf(entrada.readLine());
         imprimir_fila_columna_matriz(matint, numero);
        }
         catch (Exception exc){
              System.out.println(exc);
        }
   }
```

```
   public static void imprimir_fila_columna_matriz(int[][] mat, int
   numero){
         int fila = 0;
         int columna = MAXCOLUMNA;
         while ((fila < MAXFILA) && (columna == MAXCOLUMNA)){
              columna = obtener_pos_arreglo(mat[fila],numero);
              if (columna == MAXCOLUMNA){
                           fila++;
             }
         }
         if ((fila < MAXFILA) && (columna < MAXCOLUMNA)){
              System.out.println(numero+" se encuentra en "+fila+" y
   "+columna);
         }
         else {
              System.out.println(numero+" no se encuentra en la
   matriz");
         }
   }

   public static int obtener_pos_arreglo(int [] arr, int numero){
         int posicion = 0;
         while ((posicion < MAXCOLUMNA) && (arr[posicion] != numero)){
              posicion++;
         }
         return posicion;
   }
}
```

Ejemplo 6

```
/*Hacer un programa que dado una matriz de enteros de tamano 4*5 que se encuentra precargada, solicite al usuario una posicion fila, columna, y realice un
   corrimiento a derecha. Ademas imprima la matriz antes y despues del corrimiento
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_8_Ejemplo_6 {
   public static final int MAXFILA = 4;
   public static final int MAXCOLUMNA = 5;
   public static void main(String[] args) {
         int [][] matint;
         int fila, columna;
         matint = new int[MAXFILA][MAXCOLUMNA];
         cargar_matriz_aleatorio_int(matint);
         imprimir_matriz_int(matint);
         BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
         try{
              System.out.println("Ingrese una fila :");
              fila = Integer.valueOf(entrada.readLine());
              System.out.println("Ingrese una columna :");
              columna = Integer.valueOf(entrada.readLine());
              if ((0<=fila)&&(fila < MAXFILA)&&(0<=columna)&&(columna < MAXCOLUMNA)){
                           corrimiento_der_fila_columna(matint[fila],columna);
                           imprimir_matriz_int(matint);
              }
         }
         catch(Exception exc){
              System.out.println(exc);
         }
   }

   public static void corrimiento_der_fila_columna(int [] arrenteros, int pos){
         int indice = MAXCOLUMNA-1;
         while (indice > pos){
              arrenteros[indice] = arrenteros[indice-1];
              indice--;
        }
    }
}
```

Ejemplo de una corrida:

|3|8|4|7|5|
|4|9|6|1|7|
|1|1|1|6|2|
|5|5|7|1|2|
Ingrese una fila :
1
Ingrese una columna :
2
|3|8|4|7|5|
|4|9|6|6|1|
|1|1|1|6|2|
|5|5|7|1|2|

Práctico – primera parte

1. Hacer un programa que dado una matriz de enteros de tamaño 5*10 que se encuentra precargada, invierta el orden del contenido por fila. Este intercambio no se debe realizar de manera explícita, hay que hacer un método que incluya una iteración de intercambio.

2. Hacer un programa que dado una matriz de enteros de tamaño 5*10 que se encuentra precargada, obtenga la cantidad de números pares que tiene y la imprima.

3. Hacer un programa que dado una matriz de enteros de tamaño 5*10 que se encuentra precargada, solicite al usuario una posición fila, columna y realice un corrimiento a izquierda.

4. Hacer un programa que dado una matriz de enteros de tamaño 5*10 que se encuentra precargada, solicite al usuario un numero entero y una posición fila, columna. Con estos datos tendrá que realizar un corrimiento a derecha (se pierde el último valor en dicha fila) y colocar el numero en la matriz en la posición fila, columna indicada.

5. Hacer un programa que dado una matriz de enteros de tamaño 5*10 que se encuentra precargada, solicite al usuario un numero entero y elimine la primer ocurrencia de numero en la matriz (un número igual) si existe. Para ello tendrá que buscar la posición y si está, realizar un corrimiento a izquierda y no continuar buscando.

6. Hacer un programa que dado una matriz de enteros de tamaño 5*10 que se encuentra precargada, solicite al usuario un numero entero y elimine todas las ocurrencia de numero en la matriz si existe. Mientras exista (en cada iteración tiene que buscar la posición fila y columna) tendrá que usar dicha posición para realizar un corrimiento a izquierda.

Objetivos del tema

- Realizar operaciones de ordenamiento, eliminación, e inserción
- Incorporar el concepto de arreglo de secuencias y realizar operaciones sobre el mismo

Matrices ordenadas

- Una matriz puede estar ordenada en cada una de sus filas, en cada una de sus columnas, o toda la matriz en un sentido; por ejemplo para cada fila,columna en el siguiente sentido:

0,0; … ;0,MAXCOLUMNAS-1;1,0;…;1,MAXCOLUMNAS-1;… ;MAXFILAS-1,0;…;MAXFILAS-1,MAXCOLUMNAS-1.

- **Este último caso en esta cursada no se va considerar**, pero sirve para pensar en todas los posibles sentidos de orden.
- Vamos a trabajar con ejemplos ordenados por filas y por columnas

Ordenado por fila:
|4|5|6|9|9|
|1|4|5|7|7|
|1|1|3|4|6|
|1|2|2|6|8|

Ordenado por columna:
|4|4|2|1|5|
|6|5|4|2|7|
|8|5|5|7|8|
|9|7|5|10|9|

Métodos de ordenamiento

- Para ordenar una matriz por fila o por columna vamos aplicar los mismos algoritmos/técnicas que aplicamos para arreglos: selección, inserción, burbujeo.

- Para ordenar por filas directamente aplicamos el mismo método de arreglo. Por ejemplo para la fila de matint:

  ordenar_arreglo_seleccion(matint[i]);

- Para ordenar por columnas no se puede reutilizar los métodos de arreglos.

Ejemplo 7: ordenar por columna con selección

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_8_Ejemplo_7 {
    public static final int MAXFILA = 4;
    public static final int MAXCOLUMNA = 5;
    public static void main(String[] args) {
          int [][] matint;
          int fila, columna;
          matint = new int[MAXFILA][MAXCOLUMNA];
          cargar_matriz_aleatorio_int(matint);
          imprimir_matriz_int(matint);
          BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
          try{
               System.out.println("Ingrese una columna :");
               columna = Integer.valueOf(entrada.readLine());
               if ((0<=columna)&&(columna < MAXCOLUMNA)){
                              ordenar_matriz_columna_seleccion(matint,columna);
                              imprimir_matriz_int(matint);
               }
          }
          catch(Exception exc){
               System.out.println(exc);
          }
    }
    public static void ordenar_matriz_columna_seleccion(int [][] mat, int columna) {
          int pos_menor, tmp;
          for (int i = 0; i < MAXFILA; i++) {
               pos_menor = i;
               for (int j = i + 1; j < MAXFILA; j++){
                              if (mat[j][columna] < mat[pos_menor][columna]) {
                                             pos_menor = j;
                              }
               }
               if (pos_menor != i){
                              tmp = mat[i][columna];
                              mat[i][columna] = mat[pos_menor][columna];
                              mat[pos_menor][columna] = tmp;
               }
          }
    }
}
```

Ejemplo de una corrida:

|9|5|3|5|2|
|3|9|9|9|6|
|5|2|3|6|9|
|8|1|3|9|7|
Ingrese una columna :
1
|9|1|3|5|2|
|3|2|9|9|6|
|5|5|3|6|9|
|8|9|3|9|7|

Práctico – segunda parte

No hay que aplicar un método de ordenamiento cuando indica que la matriz se encuentra ordenada.

7. Hacer un programa que dado una matriz ordenada creciente por filas de enteros de tamaño 4*5 que se encuentra precargada, solicite al usuario un numero entero y una fila, y luego inserte el numero en la matriz en la fila indicada manteniendo su orden.

8. Hacer un programa que dado una matriz ordenada creciente por filas de enteros de tamaño 4*5 que se encuentra precargada, solicite al usuario un numero entero y una fila, y elimine la primer ocurrencia de numero en la fila indicada (un número igual) si existe.

9. Hacer un programa que dado una matriz de enteros de tamaño 4*5 que se encuentra precargada, solicite al usuario el ingreso de una fila y dos números enteros (columnas de la matriz), y ordene de forma creciente la matriz en la fila indicada entre las dos posiciones columnas ingresadas.

Matrices de secuencias

- Al igual que con los arreglos, las matrices de secuencias también sirven para representar estructuras dentro de una estructura. Por ejemplo, patrones de colores dentro de una foto (la foto es una matriz de números que hacen referencia a colores). Si detectamos los colores verdes de las hojas podríamos limpiar el fondo de la imagen.

- En este caso solo vamos a trabajar con matrices de secuencias de números enteros y matrices de secuencias de caracteres, considerando que estas secuencias solo aparecen en las filas (**no habrá secuencias por columnas**).

Matrices de secuencias

- Los métodos para trabajar con estas estructuras implican recorrer secuencias, procesar las secuencias, incorporar secuencias, y eliminar secuencias.
- Como con arreglos, vamos a partir de una solución para cargar de forma aleatoria una matriz de secuencias por fila, que nos va permitir utilizarlo para probar nuestros ejercicios.
- A continuación se dejan ejemplos de códigos (carga e impresión) para reutilizar. Los mismos se basan en las soluciones de arreglos.

Ejemplo 8

```
import java.util.Random;
public class Clase_8_Ejemplo_8 {
   public static final int MAXFILA = 4;
   public static final int MAXCOLUMNA = 20;
   public static final int MAXVALOR = 9;
   public static final int MINVALOR = 1;
   public static final double probabilidad_letra = 0.4;
   public static final double probabilidad_numero = 0.4;
   public static void main(String[] args) {
    char [][] matchar;
    int [][] matint;
    matchar = new char[MAXFILA][MAXCOLUMNA];
    matint = new int[MAXFILA][MAXCOLUMNA];
    cargar_matriz_aleatorio_secuencias_char(matchar);
    imprimir_matriz_char(matchar);
    cargar_matriz_aleatorio_secuencias_int(matint);
    imprimir_matriz_int(matint);
    }

   public static void imprimir_matriz_char(char [][] mat){
    for (int fila = 0; fila < MAXFILA; fila++){
         imprimir_arreglo_secuencias_char(mat[fila]);
         System.out.println("");
    }
    }

   public static void imprimir_matriz_int(int [][] mat){
    for (int fila = 0; fila < MAXFILA; fila++){
         imprimir_arreglo_secuencias_int(mat[fila]);
         System.out.println("");
    }
    }

   public static void cargar_matriz_aleatorio_secuencias_int(int [][]
   mat){
    for (int fila = 0; fila < MAXFILA; fila++){
         cargar_arreglo_aleatorio_secuencias_int(mat[fila]);
    }
    System.out.println("");
    }

   public static void cargar_matriz_aleatorio_secuencias_char(char
   [][] mat){
    for (int fila = 0; fila < MAXFILA; fila++){
         cargar_arreglo_aleatorio_secuencias_char(mat[fila]);
    }
    System.out.println("");
    }
}
```

Ejemplo de una corrida:

```
|  |  |r|m|o|p|  |q|  |i|m|e|a|  |i|z|  |  |d|  |
|  |  |  |f|  |w|y|i|x|l|p|a|r|  |o|  |  |  |  |  |
|  |j|  |w|  |o|  |  |x|g|  |t|g|e|v|  |q|d|  |  |
|  |m|y|  |c|  |  |l|  |e|d|q|  |  |  |v|  |x|f|  |

|0|6|6|0|4|4|0|0|3|9|0|8|5|7|0|6|6|0|1|0|
|0|3|0|5|0|9|0|5|0|0|0|0|3|0|9|3|4|0|0|0|
|0|2|0|0|0|7|0|0|7|0|4|0|0|0|2|0|8|4|5|0|
|0|5|0|3|1|9|0|5|7|0|0|4|0|0|0|9|9|0|0|0|
```

Práctico - tercera parte

Se tiene una matriz de enteros de tamaño 4*20 de secuencias de números entre 1 y 9 (por cada fila), separadas por 0. La matriz esta precargada, y además cada fila empieza y termina con uno o más separadores 0.

Además, se tiene una matriz de caracteres de tamaño 4*20 de secuencias de caracteres letras minúsculas entre 'a' y 'z' (por cada fila), separadas por ' ' (espacios). La matriz esta precargada, y además cada fila empieza y termina con uno o más separadores ' '.

Considere para los siguientes ejercicios estos tipos de matriz.

10. Hacer un programa que dada la matriz de secuencias de enteros definida y precargada, permita obtener a través de métodos la posición de inicio y la posición de fin de la secuencia ubicada a partir de una posición entera y una fila, ambas ingresadas por el usuario. Finalmente, si existen imprima por pantalla ambas posiciones obtenidas.

Ejemplo 9

```
/*Hacer un programa que dado la matriz definida y precargada, imprima lo que suma el
  contenido de cada secuencia.
 */
import java.util.Random;
public class Clase_8_Ejemplo_9 {
  public static final int MAXFILA = 4;
  public static final int MAXCOLUMNA = 20;
  public static final int MAXVALOR = 9;
  public static final int MINVALOR = 1;
  public static final double probabilidad_numero = 0.4;
  public static void main(String[] args) {
  int [][] matint;
  matint = new int[MAXFILA][MAXCOLUMNA];
  cargar_matriz_aleatorio_secuencias_int(matint);
  imprimir_matriz_int(matint);
  imprimir_suma_secuencias_matriz(matint);
  }

  public static void imprimir_suma_secuencias_matriz(int[][] mat){
  for (int fila = 0; fila < MAXFILA; fila++){
      System.out.println("Para la fila "+fila);
      imprimir_suma_cada_secuencia(mat[fila]);
  }
  }
}
```

Práctico - tercera parte

11. Hacer un programa que dada la matriz de secuencias de enteros definida y precargada permita encontrar por cada fila la posición de inicio y fin de la secuencia cuya suma de valores sea mayor.

12. Hacer un programa que dada la matriz de secuencias de caracteres definida y precargada, permita encontrar por cada fila la posición de inicio y fin de la anteúltima secuencia (considerar comenzar a buscarla a partir de la ultima posición de la fila).

13. Hacer un programa que dada la matriz de secuencias de enteros definida y precargada, y un número entero ingresado por el usuario, elimine de cada fila las secuencias de tamaño igual al número ingresado.

14. Hacer un programa que dada la matriz de secuencias de caracteres definida y precargada, elimine de cada fila todas las ocurrencias de una secuencia patrón dada por un arreglo de caracteres de tamaño igual al tamaño de columnas de la matriz (solo tiene esa secuencia con separadores al inicio y al final). Al eliminar en cada fila se pierden los valores haciendo los corrimientos.

15. Hacer un programa que dada la matriz de secuencias de caracteres definida y precargada elimine todas las secuencias que tienen orden descendente entre sus elementos.

Práctico - tercera parte

16. Hay dos matrices MAT1 y MAT2 de secuencias de caracteres letras separados por espacios de tamaño MAXF x MAXC que están precargadas. Ambas matrices están precargadas y cada fila empieza y termina con caracteres espacios. Además se tiene el siguiente método:
    - un método que retorna el índice inicial de la secuencia de mayor tamaño de un arreglo de secuencias (de caracteres letras minúsculas separados por espacios) de tamaño MAXFIL.

    Se pide realizar un programa que

    - contenga la definición de los encabezados de los métodos de carga de la matriz y del método mencionado en el enunciado (se supone que existen y no se requiere implementarlos).
    - para MAT1 y MAT2 elimine de cada secuencia el primer carácter vocal.
    - para MAT1 agregue al principio de cada secuencia el primer carácter de la secuencia de mayor tamaño de dicha fila.
    - en cada fila, si se verifica que la secuencia de mayor tamaño de la fila para MAT1 es mayor que la primer secuencia en dicha fila para MAT2, las intercambie (la que está en MAT1 pasa a MAT2 y la que esta en MAT2 pasa a MAT1) sin usar estructuras auxiliares (otros arreglos o matrices).
    - para un valor de fila ingresado por el usuario verifique e imprima si la primera secuencia de MAT1 en dicha fila es igual la primera secuencia de MAT2 en dicha fila.

Indexación de matrices usando arreglos

- De forma similar a la indexación de arreglos podemos tener situaciones donde la información o los datos está en una matriz, y se tienen arreglos que permiten acceder a los índices de la matrices (filas o columnas) con diferentes criterios.
  - Por ejemplo, se tiene una matriz A de enteros. Un arreglo B tiene los índices de las filas de A, que permite mantener el un orden creciente entre las filas de la matriz por la suma de sus filas. Un arreglo C tiene los índices de las columnas de A cuya suma es par (-1 para completar si no hay).

A= **|3|1|2|4|3|**
|2|5|2|1|5|
|1|1|3|3|2|

B= **|2|0|1|**

C= **|0|3|4|-1|-1|**

- Siempre hay que recordar que la información de la estructura base (A) no se replica ni se modifica para tener otro orden u otra información sobre su contenido (salvo que se diga expresamente), sino que se generan índices para acceder de la forma requerida.

Indexación de matrices usando arreglos

- ¿Cómo se accede a la información desde la indexación?:
  - B y C tienen índices filas y columnas de A respectivamente, si queremos por ejemplo imprimir la información de A según B y C haremos:

```
for(int i = 0, i<MAX_FILAS, i++){
    System.out.println(obtener_suma_fila(A,B[i]);
    //i es índice de B
    //B[i] es un índice fila de A
    //obtener_suma_fila permite obtener lo que suma la fila B[i] de A
}


int i = 0;
while ((i<MAX_COLUMNAS)&&(C[i]!=-1)){
    System.out.println(obtener_suma_columna(A,C[i]);
    i++;
    //i es índice de C
    //C[i] es un índice columna de A
    //obtener_suma_columna permite obtener lo que suma la columna C[i] de A
}
```

Ejemplo 10

- Dada una matriz MAT de enteros precargada, se pide cargar un arreglo ARR_SUMAS_FILAS_ORD los índices de las filas de MAT y ordenarlo de forma tal que permita mantener un orden creciente de MAT por la suma de sus filas.

  Link solución Clase_8_Ejemplo_10.java

Práctico - cuarta parte

17. Para el Ejemplo 10, hacer:
   - cargar un arreglo ARR_SUMAS_COLUMNAS_PARES los índices de las columnas de MAT cuya suma es par (-1 para completar si no hay).
   - ordenar ARR_SUMAS_COLUMNAS_PARES de forma ascendente según las sumas de columnas de MAT utilizando un método de ordenamiento.

18. Se tiene una matriz MAT de caracteres letras minúsculas precargada, se pide:
   - cargar un arreglo ARR_CANT_VOCALES_FILAS_ORD los índices de las filas de MAT y ordenarlo de forma tal que permita mantener un orden decreciente de MAT por la cantidad de vocales de sus filas.
   - dada una fila ingresada por el usuario desde teclado, eliminar la fila de la matriz MAT y actualizar el arreglo ARR_CANT_VOCALES_FILAS_ORD (no hay que aplicar método de ordenamiento).