Estructuras de Datos.
Facultad de Ciencias, UNAM 2015-2.
Práctica 03. Matrices

Armando Ballinas Nangüelú.
armballinas@gmail.com

Fecha de entrega: 05 de marzo de 2015

1. Introducción.

En esta práctica se realizarán dos implementaciones del tipo de dato *matriz*. La primera será mediante un arreglo de arreglos y la segunda mediante un polinomio de direccionamiento. Además que se debe tener consideración del espacio que ocupan ambas representaciones.

2. Representación de matrices.

Las matrices serán leídas de un archivo XML con un DTD específico. Se incluyen las clases que realizan esta función en Java. La definición de una matriz está especificada en el archivo *matriz.dtd*.

La clase LectorMatriz.java permite leer las entradas de una matriz especificada en un archivo .xml. La clase ManejadorMatriz.java provee la clase para poder extraer los datos del archivo XML de manera que satisfaga el archivo matriz.dtd. El alumno debe leer y analizar el contenido de estas clases para poder utilizarlas.

3. Tipos de matrices.

En la práctica habrá que almacenar varios tipos de matrices, estos distintos tipos están definidos en la clase ManejadorMatriz.java. Entre estos se encuentran tipos para definir matrices simétricas o triangulares.

El objetivo de definir estas constantes es que en el almacenamiento de la matriz no se desperdicie espacio en entradas innecesarias o redundantes, es decir, si la matriz es simétrica solo se debe almacenar una mitad, ya sea la superior o la inferior. Si la matriz es triangular inferior sin diagonal, solo se deben almacenar las entradas con elementos válidos.

4. Implementaciones de matrices.

En la práctica se debe implementar la interfaz Matriz de dos maneras diferentes. En ambos casos solo se deben almacenar las entradas que no sean redundantes.

4.1. Arreglo de arreglos.

En la primera implementación las matrices se deben almacenar en un arreglo de arreglos. El primer arreglo representará cada uno de los renglones de la matriz. Cada elemento de este arreglo será un arreglo que represente a las columnas de la matriz.

Esta clase debe llamarse `MatrizArreglo.java`

4.2. Polinomio de direccionamiento.

La segunda implementación almacenará todos los elementos en un arreglo contiguo. Y utilizará un polinomio de direccionamiento para indexar los elementos.

Esta clase debe llamarse `MatrizPolinomio.java`

4.3. Clase Abstracta.

Dado que ambas implementaciones deben implementar las mismas operaciones, se puede crear una clase abstracta que implemente aquellos métodos que sean comunes a ambas implementaciones y hacer que `MatrizArreglo` y `MatrizPolinomio` hereden de esta.

4.4. Clase de Prueba.

Se debe definir una clase `Main.java` en donde se creará un programa de prueba en donde se realizará un menú para que el usuario interactúe con el programa.

5. Entrada y Salida.

5.1. Entrada.

La entrada del programa debe hacerse por medio de un menú interactivo pidiendo al usuario la operación a realizar así como los archivos que contienen las especificaciones de las matrices. En el caso de la operación de multiplicación por un escalar, este se debe pedir al usuario por medio de la entrada estándar.

5.1.1. Menú interactivo.

Una vez que se ejecuta el programa. se debe mostrar un menú donde primero se le debe pedir al usuario qué tipo de implementación de matrices quiere usar ya sea basadas en arreglos de arreglos o basada en un polinomio de direccionamiento.

Una vez que el usuario haya elegido la implementación deseada se deben mostrar una serie de opciones. Cada opción contendrá una operación a realizar. El usuario debe elegir una operación para ser realizada, una vez hecho esto se deben solicitar la ruta de los archivos que contienen a las matrices. En este punto se utilizarán las clases descritas al inicio de la práctica para poder extraer la información de estos archivos y crear las matrices correspondientes. Como ya se mencionó, si el usuario escoge la operación de producto por escalar entonces se debe pedir solo un archivo de matriz y después un número de punto flotante.

Una vez realizada la operación se mostrará el resultado y se le pedirá al usuario si desea realizar otra operación o finalizar el programa.

5.2. Salida.

La salida será por la salida estándar y el programa mostrará el resultado de la operación escogida por el usuario. Primero mostrará los argumentos de la operación y después la matriz resultante.

6. Ejercicios.

1. (8 pts) Realizar las implementaciones de `MatrizArreglo` y `MatrizPolinomio`
2. (2 pts) Realizar el menú y programa de prueba `Main`

7. Puntos extra.

1. (Hasta 3 puntos extra) Agregar una opción en el menú para guardar la matriz resultante de una operación que devuelva un objeto `Matriz`. Esta matriz se debe guardar en un archivo .xml respetando el DTD `matriz.dtd`.

8. Archivos a entregar.

Se deben entregar al menos los siguientes archivos:

1. `Matriz.java` El archivo con la interfaz del TDA Matriz.
2. `MatrizArreglo.java` El archivo que contiene la clase con la implementación de la matriz utilizando un arreglo de arreglos.
3. `MatrizPolinomio.java` El archivo que contiene la clase con la implementación de la matriz utilizando un polinomio de direccinamiento.
4. `Main.java` El archivo que contiene la clase Main, es decir, el archivo que tiene el programa de prueba y el menú interactivo.
5. `Readme.txt` Un archivo donde incluirán su nombre, número de cuenta y correo electrónico. Así como la manera de compilar y ejecutar su programa.