



Universidad Autónoma de Zacatecas

Unidad Académica de Ingeniería Eléctrica

Programa Académico de Ingeniería de Software

Práctica 8

Datos generales:

Nombre de la Práctica	Arreglos multidimensionales
Nombre de la carrera	Ingeniería de Software
Nombre de la materia	Laboratorio de Estructuras de Datos
Número y nombre de Unidad(es) temática(s)	1. Introducción a las estructuras de datos y estructuras fundamentales.
Docente que imparte la materia	Aldonso Becerra Sánchez
Fecha de entrega para los alumnos	3-septiembre-2021 7:00 am
Fecha de entrega con extensión y penalización	4-septiembre-2021 7:00 am
Fecha de elaboración	2-septiembre -2021

Objetivo de la Práctica	Comprender el uso de los arreglos multidimensionales como TDA.
Tiempo aproximado de realización	3 horas
Introducción	La facilidad que los arreglos multidimensionales tienen para permitir guardar datos en forma de columnas/renglones, los hace pertinentes para la resolución de muchos problemas donde se requiere esta situación. El único detalle con esta cuestión es que es poco flexible el número de elementos que podemos manipular, ya que se requiere conocer a priori la cantidad de elementos a guardar.



Universidad Autónoma de Zacatecas

Unidad Académica de Ingeniería Eléctrica

Programa Académico de Ingeniería de Software

Referencias que debe consultar el alumno (si se requieren):

Referencia 1:

1. Cairo, Osvaldo; Guardati, Silvia. Estructura de Datos, Tercera Edición. McGraw-Hill, México, Tercera Edición, 2006.

Referencia 2:

2. Mark Allen Weiss. Estructura de datos en Java. Ed. Addison Wesley.

Referencia 3:

3. Joyanes Aguilar, Luis. Fundamentos de Programación. Algoritmos y Estructuras de Datos. Tercera Edición, 2003. McGraw – Hill.

Actividades que debe realizar el alumno:

Actividad inicial:

Lea primero toda la práctica. No inicie a programar sin leer todo cuidadosamente primero. Recuerde que debe generar el reporte en formato IDC.

Actividad 1:

Primero genere la **Introducción**.

Actividad 2:

Defina un TDA llamado “Tabla2D”, que permita realizar las siguientes operaciones:

1. Constructor: Tabla2D(int filas, int columnas).
2. Constructor: Tabla2D (int filas, int columnas, Object contenido). El contenido es el valor que debe inicializar toda la tabla con este valor.
3. Leer un elemento de la matriz: Object obtenerCelda(int fila, int columna).
4. Cambiar un elemento: boolean asignarCelda(int fila, int columna, Object contenido).
5. Obtener el número de elementos en renglones y columnas: int getFilas(), int getColumnas().
6. Imprimir los datos de la matriz: void imprimirR(). Imprime la matriz en formato renglón por renglón.
7. Imprimir los datos de la matriz: void imprimirC(). Imprime la matriz en formato columna por columna.
8. Hacer la transpuesta de la matriz: void transpuesta().



Universidad Autónoma de Zacatecas

Unidad Académica de Ingeniería Eléctrica

Programa Académico de Ingeniería de Software

9. Inicializar todos los elementos de la matriz: void rellenar(Object contenido). Validar si tiene
10. Generar y regresar una “copia” de la tabla: Tabla2D clonar().
11. Generar un método que diga si el contenido de dos tablas es igual: boolean esIgual(Tabla2D matriz2). Posición por posición como contenido.
12. Defina un método que genere un vector columna con los datos especificados por el usuario, a partir de la tabla vacía: boolean vectorColumna(int numFilas, Object contenido).
13. Defina un método que genere un vector renglón con los datos especificados por el usuario, a partir de la tabla vacía: boolean vectorRenglon(int numColumnas, Object contenido).
14. Defina un método que permita crear/redimensionar/substituir la tabla actual por una pasada como argumento. boolean redefinirTabla(Tabla2D tabla2). El tamaño de la tabla actual deberá quedar igual al que se pasa como argumento.
15. Defina un método que permita agregar un ArregloDatos pasado como argumento como renglón de la tabla existente. Para esto deberá validar los tamaños del arreglo que cumplan con el cometido de la tabla. boolean agregarFila(ArregloDatos arreglo).
16. Defina un método que permita agregar un Arreglo pasado como argumento como columna de la tabla existente. Para esto deberá validar los tamaños del arreglo que cumplan con el cometido de la tabla. boolean agregarColumna(ArregloDatos arreglo).
17. Defina un método que permita agregar una tabla nueva a la tabla actual. Para esto deberá validar que sea como elementos agregados en columnas (hacia la derecha). boolean agregarTablaxColumna(Tabla2D tabla2).
18. Defina un método que permita agregar una tabla nueva a la tabla actual. Para esto deberá validar que sea como elementos agregados en filas (hacia abajo). boolean agregarTablaxRenglon(Tabla2D tabla2).
19. Defina un método que permita pasar una tabla 2D a tabla 3D. Para eso deberá llevar la matriz 2D actual más un argumento que es un arreglo de matrices. public Tabla3D aTabla3D(ArregloDatos tablas). la variable tablas contiene las tablas (la tercera dimensión, profundidad) que permitirán agregarse a la tabla actual 2d y generar una tabla 3D, la cual deberá ser regresada. No olvide validar las dimensiones.
20. Pasar una tabla 2D a vector columna. Tabla2D aVectorColumna(). Este método convierte la matriz actual acomodando cada columna una debajo de otra para formar un vector columna.
21. Pasar una tabla 2D a vector renglón. Tabla2D aVectorRenglón(). Este método convierte la tabla actual acomodando cada renglón uno enseguida del otro (hacia la derecha) para formar un vector renglón.



Universidad Autónoma de Zacatecas

Unidad Académica de Ingeniería Eléctrica

Programa Académico de Ingeniería de Software

22. Eliminar una columna de una tabla boolean quitarColumna(TipoColumna tipoCol). TipoColumna es un enumerado para columna izquierda o derecha.
23. Eliminar una fila de una tabla boolean quitarFila(TipoRenglon tipoFila). TipoFila es un enumerado para fila superior o inferior.
24. Eliminar una fila de una posición de una matriz boolean equitarFila(int indice). “indice” indica la fila a eliminar.
25. Eliminar una columna de una posición de una matriz boolean quitarColumna(int indice). “indice” indica la columna a eliminar.

Haga el programa (actividad 2, la cual es el **Desarrollo** del reporte).

Actividad 3:

Pruebe el funcionamiento del programa de la actividad 2 con todo y sus capturas de pantalla.

Actividad 4:

Realice la sección de **Código agregado** (diagrama de clases UML).

Actividad 5:

Realice la sección de **Pre-evaluación** (use los lineamientos establecidos).

Actividad 6:

Finalmente haga las **Conclusiones**.

Actividad 5:

Enviar en <http://ingsoftware.reduaz.mx/moodle>

Archivo anexo que se requiere para esta tarea (opcional):

Dudas o comentarios: a7donso@gmail.com