Universidad Autónoma de Zacatecas

Unidad Académica de Ingeniería Eléctrica

Programa Académico de Ingeniería de Software.

Nombre de la Práctica Árboles.

Numero de Práctica 27.

Nombre de la carrera Ingeniería de Software.

Nombre de la materia Lab. Estructuras de Datos.

Nombre del alumno Jesús Manuel Juárez Pasillas.

Nombre del docente Aldonso Becerra Sánchez.

Fecha: 03/11/2021.

Práctica 27: Árboles.

Introducción:

Los montículos son arboles donde se tiene definido un “orden” ya que para que cumpla con la característica de un montículo el nodo raíz tiene que ser mayor o menor que sus nodos hijos según se defina. Los nodos que se van agregando siempre se agregan al final y de ahí se tiene que ir comparando con su padre para que no rompa el “orden” en el que se esta agregando los nodos, si estos nodos rompen dicho orden se intercambian con sus padres iterativamente hasta encontrar su lugar.

Desarrollo:

Para realizar esta practica se tuvo que hacer un nuevo tipo de nodo. Este nuevo nodo será un nodo el cual almacene los hijos izquierda y derecha de nodo, además del nodo padre y un objeto donde se almacena el elemento del nodo.

Esta estructura de datos se agrego en el paquete “estructuraslineales.registros” y se le llamo NodoTripe.

En esta clase se agregaron los métodos getters y setters, además de su toString.

- Obtiene el padre del nodo.
  public NodoTriple getPadre().
- Cambia el padre del nodo.
  public void setPadre(NodoTriple padre).
- Obtiene el elemento del nodo.
  public Object getElemento().
- Cambia el elemento del nodo.
  public void setElemento(Object elemento).
- Obtiene el hijo izquierdo del nodo.
  public NodoTriple getHijoIzq().
- Cambia el nodo izquierdo del nodo.
  public void setHijoIzq(NodoTriple hijoIzq).
- Obtiene el hijo derecho del nodo.
  public NodoTriple getHijoDer().

- Cambia el hijo derecho del nodo por el pasado.
  public void setHijoDer(NodoTriple hijoDer).

- Obtiene el toString del elemento.
  public String toString().

Se uso esta estructura para realizar la estructura de datos MonticuloArbol, el cual será un árbol que se comportará como un montículo.

Esta clase se creo dentro del paquete “estructurasnolineales” y se le llamo MonticuloArbol, en esta clase se agregaron los atributos de raíz (NodoTriple) y orden (TipoOrden) y los métodos de vacío, agregar, quitar e imprimir el árbol recorriéndolo en forma de amplitud.

- Determina si el árbol esta vacío o no.
  public boolean vacio().

- Agrega un elemento al final del árbol, luego acomoda ese elemento agregado con forme al orden establecido para el montículo.
  public boolean agregar(Object elemento).

- Elimina el último elemento del árbol.
  public Object quitar().

- Imprime el montículo haciendo el recorrido por amplitud.
  public void amplitud().

Con un árbol binario no se pudo realizar estos métodos debido a la estructura de sus nodos que no permitían obtener los padres de cada nodo, es por ello que se opto por realizar el nuevo nodo triple. Con este nuevo nodo es mucho más fácil realizar los procedimientos con los cuales realizar las tareas pedidas (agregar y quitar que son los métodos más complicados de realizar).

Teniendo esto podemos realizar la simulación de un montículo en donde se le pasan procesos a evaluar, los cuales contienen una prioridad y con esto asignar su posición en el árbol.

Ya se tenia una clase Proceso la cual se usó para los montículos con arreglos (práctica 14), por lo que se utilizó estas clases.

Para un uso más fácil del programa, los datos generales de los procesos son totalmente automáticos, lo único que se pide es la prioridad de cada proceso que se valla agregando, luego esto imprimirá el montículo el cual solo obtendrá la prioridad de cada proceso y por último sacara los procesos con forme a su prioridad y se imprimirán.

Nota: Toda la documentación del proyecto esta agregada en la carpeta “doc” dentro de la carpeta del proyecto (“edylab_2021_27/doc”).

Capturas del programa funcionando:

Primero se pide el numero de procesos a agregar, pide las prioridades de cada proceso y lo demás es automático.

La prueba se encuentra en el paquete “pruebas” y se llama **PruebaMonticuloArbol**.

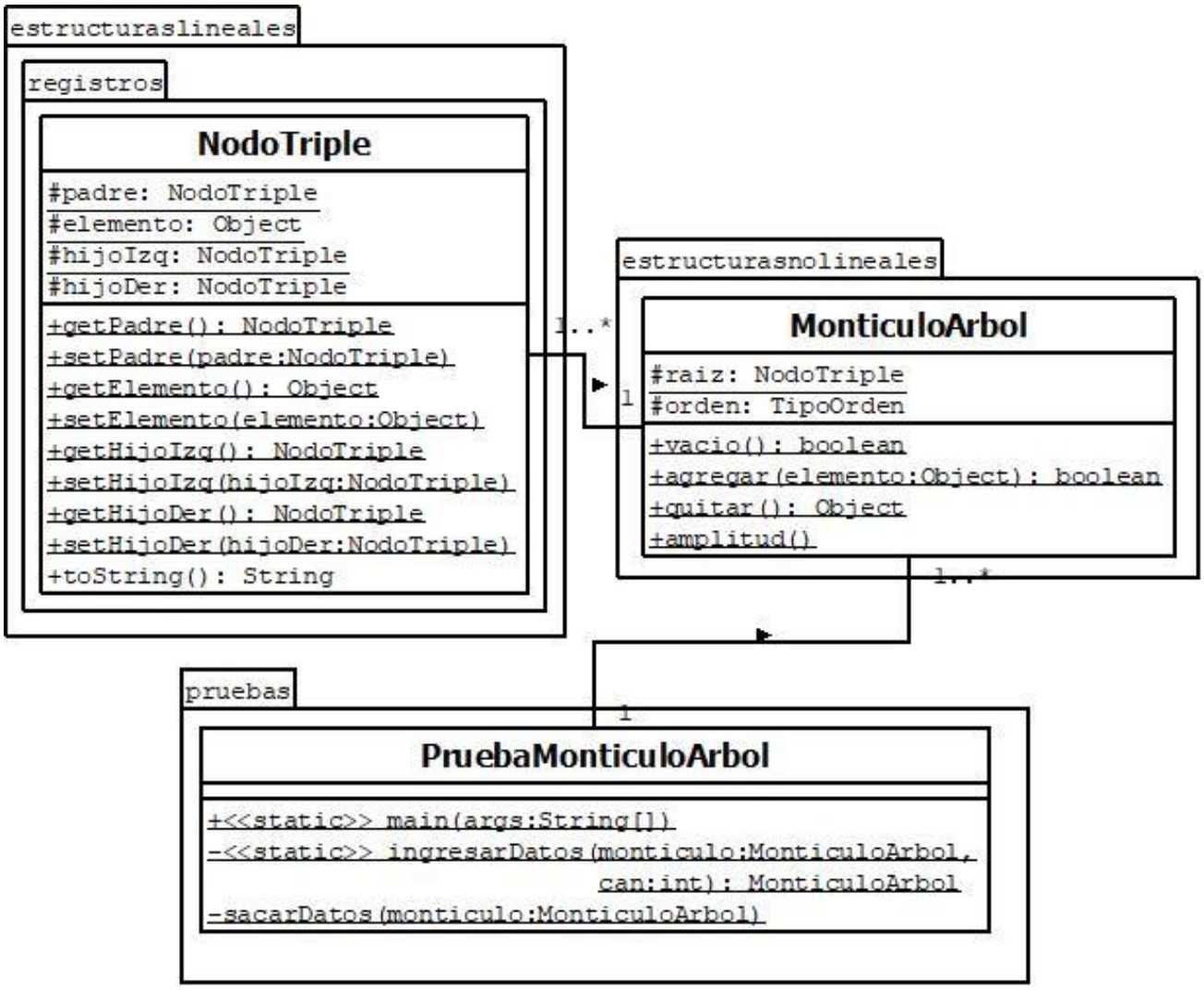
```
Impresion del monticulo recorrido por amplitud (Solo se ve la prioridad del proceso):
1 2 3 5 4
```

```
Salida de los procesos:
Proceso:
    nombre = p2
    comando = c2
    archvio = a2
    ruta = r2
    propietario = p2
    tipo = SISTEMA, 1
Proceso:
    nombre = p4
    comando = c4
    archvio = a4
    ruta = r4
    propietario = p4
    tipo = DESCARGA, 2
Proceso:
    nombre = p5
    comando = c5
    archvio = a5
    ruta = r5
    propietario = p5
    tipo = ABRIR, 4
```

```
Proceso:
    nombre = p3
    comando = c3
    archvio = a3
    ruta = r3
    propietario = p3
    tipo = GUARDADO, 3
Proceso:
    nombre = p1
    comando = c1
    archvio = a1
    ruta = r1
    propietario = p1
    tipo = CERRAR, 5
```

Código agregado:

Todas las clases y métodos que se ven en el diagrama son nuevos, solo los paquetes no son nuevos.



Pre-evaluación:

Pre-Evaluación para prácticas de Laboratorio de Estructuras de Datos	PRE-EVALUACIÓN DEL ALUMNO
CUMPLE CON LA FUNCIONALIDAD SOLICITADA.	Sí
DISPONE DE CÓDIGO AUTO-DOCUMENTADO.	Sí
DISPONE DE CÓDIGO DOCUMENTADO A NIVEL DE CLASE Y MÉTODO.	Sí
DISPONE DE INDENTACIÓN CORRECTA.	Sí
CUMPLE LA POO.	Sí
DISPONE DE UNA FORMA FÁCIL DE UTILIZAR EL PROGRAMA PARA EL USUARIO.	Sí
DISPONE DE UN REPORTE CON FORMATO IDC.	Sí
LA INFORMACIÓN DEL REPORTE ESTÁ LIBRE DE ERRORES DE ORTOGRAFÍA.	Sí
SE ENTREGÓ EN TIEMPO Y FORMA LA PRÁCTICA.	Sí
INCLUYE LA DOCUMENTACIÓN GENERADA CON JAVADOC.	Sí
INCLUYE EL CÓDIGO AGREGADO EN FORMATO UML.	Sí
INCLUYE LAS CAPTURAS DE PANTALLA DEL PROGRAMA FUNCIONANDO.	Sí
LA PRÁCTICA ESTÁ TOTALMENTE REALIZADA (ESPECIFIQUE EL PORCENTAJE COMPLETADO).	100%
Observaciones:	

Conclusión:

Los montículos nos permiten realizar una estructura de datos como si fuera una cola de prioridad ya que primero saca los elementos de mas prioridad, si tienen la misma prioridad saca los elementos que se agregaron primero y luego los demás. Esto nos permite simular la entrada y salida de procesos los cuales tienen una prioridad la cual les define un lugar en el montículo, y con esto sacar los datos conforme a la prioridad que les toco.