PRÁCTICA Nº 6: 2 sesiones

(del 12 de mayo al 23 de mayo de 2003)

Árboles binarios de expresiones

0. OBJETIVOS

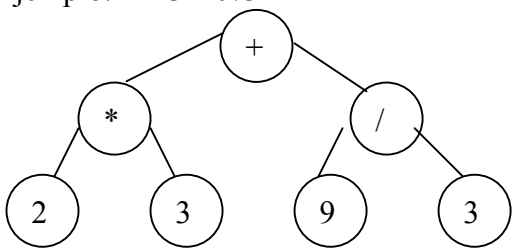
El objetivo de esta práctica es la implementación del TAD árbol binario y su uso para almacenar expresiones matemáticas en memoria.

1. INTRODUCCIÓN

Se desea crear un programa que básicamente permita introducir y evaluar expresiones matemáticas con operadores binarios, además de otras opciones. El programa presentará un menú que permita realizar las siguientes operaciones:

1. Introducir por teclado una expresión matemática de números enteros en notación infija, sin paréntesis. Los operadores reconocidos serán los siguientes: ^(elevado a), *(multiplicación), /(división), +(suma) y -(resta). Los espacios en blanco no se tendrán en cuenta.

Ejemplo: 2 * 3 + 9/3



2. Visualizar la expresión introducida en notación prefija. Ejemplo: *+ * 2 3 / 9 3*
3. Visualizar la expresión introducida en notación infija. Ejemplo: *2 * 3 + 9 / 3*
4. Visualizar la expresión introducida en notación postfija. Ejemplo: 2 3 * 9 3 / +
5. Evaluar la expresión y mostrar el resultado en pantalla. Ejemplo: *2 * 3 + 9 / 3 = 9*
6. Generar código ensamblador por pantalla. Por ejemplo para la expresión: *42 * 7 + 2 / 3* se visualizará el siguiente código:

```
div   2     3     tmp2
mul   42    7     tmp3
add   tmp3  tmp2  tmp1
```

7. Mostrar información del árbol binario:
   - Altura del árbol. Si un árbol está vacío su altura es cero, y si no está vacío su altura es uno más el máximo de las alturas de cada uno de sus hijos. Ejemplo: *3*
   - Número de hojas del árbol. Si un árbol está vacío su número de hojas es cero. Si no está vacío, hay dos posibilidades, que el árbol sea una hoja en cuyo caso vale 1, o que no lo sea por lo que se sumarán las hojas que tenga el hijo izquierdo y el hijo derecho. Ejemplo: *4*
   - Número de nodos del árbol. Si un árbol está vacío el número de nodos que tiene es cero, y si no lo está el número de nodos es 1 más la suma de nodos del hijo izquierdo más los del hijo derecho.
   - Número de nodos internos del árbol. Se calcula restando al número de nodos el número de hojas.
   - Indica si el árbol está lleno o no. Un árbol está lleno si el número de nodos es igual a 2^altura-1.

2. REALIZACIÓN DE LA PRÁCTICA

En esta práctica se debe implementar la clase árbol binario mediante punteros, que se ha visto en clase de teoría, y la clase pila utilizada en la tercera práctica. La información que almacenen ambas serán cadenas.

```
typedef string Valor;
class Arbol
{
    private:
        typedef Arbol *PunteroArbol;
        bool esvacio;
        Valor info;
        PunteroArbol izdo;
        PunteroArbol dcho;
    public:
        Arbol (void);           /*Constructor*/
        Arbol (const Arbol &);  /*Constructor de copia*/
        ~Arbol(void);           /*Destructor          */

        void HacerArbol(Arbol &, Valor, Arbol &);
        bool ArbolVacio(void);
        bool Informacion (Valor &);
        Arbol &HijoIzdo(void);
        Arbol &HijoDcho(void);
};
class Pila
{
    private:
...
    public:
        Pila (void);
        bool Apilar (Valor);
        bool Desapilar (void);
        bool CimaPila(Valor &);
        bool PilaVacia(void);
};
```

La generación del árbol binario se realizará en dos pasos como se indica en los apuntes:

1. Primero se traduce la expresión a notación postfija mediante dos pilas, quedando la expresión algebraica en una de ellas.
2. Posteriormente se genera el árbol binario.

Se realizará un programa ##**Expresiones.cpp** que permita evaluar expresiones introducidas por teclado y el resto de opciones indicadas en el menú. Este programa trabajará con objetos de la clase ArbolBinario y Pila.

La generación de código ensamblador se puede hacer mediante una función recursiva que en el caso de ser un operador visualiza: la operación, la variable temporal obtenida al generar código para la rama izquierda y la variable temporal obtenida al generar código para la rama derecha y la variable temporal, y finalmente retorna la variable temporal. Si no es un operador retorna la información que contiene el nodo.

3. ENTREGA DE PROGRAMAS

Los siguientes ficheros se podrán entregar al profesor hasta el comienzo de la siguiente sesión de prácticas:

1. Ficheros de la clase Pila (**##Pila.cpp** y **##Pila.h**).
2. Ficheros de la clase ArbolBinario (**##ArbolBinario.cpp** y **##ArbolBinario.h**).
3. Fichero con el programa principal (**##Expresiones.cpp**).

Nota Muy Importante

Antes de poder empezar a realizar cualquiera de las prácticas es necesario presentar las hojas de especificación de programas (documentación de programas) con las tareas que se van a realizar en la práctica, explicando brevemente como se van a solucionar los problemas que se plantean.

ENTREGA DE PROGRAMAS: Al comenzar la sesión de prácticas del 26 al 30 de Mayo.