Estructuras de Datos y de la Información

Práctica 7

Nodo del Árbol Multicamino

OBJETIVO: En esta práctica se implementa la estructura de nodo de un árbol multicamino de orden m, AMC(m).

FECHA: La práctica se presentará entre los días 1 y 17 de marzo de 2005.

ENUNCIADO: Desarrolla un programa en lenguaje C++ que lea una secuencia de `N` números enteros y los almacene en un objeto de tipo `NodeAMC<int, N>`. A continuación, el programa mostrará la secuencia ordenada por pantalla.

Ejemplo:

```
Introduce número de enteros. N = 5
Introduce secuencia de 5 enteros: 3 5 2 8 1
Secuencia ordenada: 1 2 3 5 8
```

NOTAS DE IMPLEMENTACION:

Cada nodo del AMC(m) es un registro de longitud fija que contiene la información del `struct TNodo` visto en las transparencias del tema 6. En lugar de la definición del `struct TNodo`, se implementa una plantilla `NodeAMC<class KEY, TSize m>`, donde el parámetro `KEY` define el tipo de las claves almacenadas; y el parámetro `m` define el número máximo de enlaces del nodo.

```
template <class KEY, TSize m>
class NodeAMC {
  private:
    KEY K[m-1];
    Tpointer A[m];
    TSize nK;

  public:
    // Constructores
    NodoAMC() {nk = 0; for(TSize i = 0; i < m; A[i++] = -1);}
    NodoAMC(const KEY& k) {for(nk=1, K[0]=k, TSize i=0; i < m; A[i++]=-1);}

    // Métodos para acceder a los campos privados
    KEY& operator[](const int i) {return K[i-1];}
    KEY operator[](const int i) const {return K[i-1];}
    TPointer& link(const TSize i) {return A[i];}
    TSize nk() {return nK;}

    // Método de búsqueda binaria en el vector de claves. Si lo encuentra,
    // devuelve la posición; sino devuelve el enlace al siguiente nodo.
    bool Search(const KEY& k, TSize& pos);

    // Método de inserción en orden de una clave y un puntero. Los valores
    // de clave no se repiten. Incrementa el contador de claves nK.
    void Insert(const KEY& k, const TPointer l);

    // Método de eliminación de una clave y un puntero. Decrementa el
    // contador de claves nK.
    void Remove(const KEY& k);

    // Métodos utilizados para almacenar en un FileBuf.
    TSize Pack(IOBuffer* buf);
    TSize UnPack(IOBuffer* buf);
};
```