Practica 2 – EDA 2
GRUP15

Objetivo a resolver

1. Crear un diccionario que implemente un árbol binario.

2. Cada palabra nueva a insertar recorre el árbol buscando su posición adecuada, respeto a su valor ASCII de palabra.

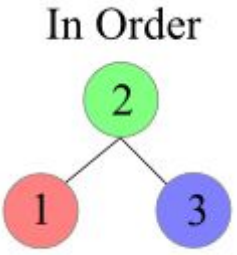
3. Para encontrar su posición se llama a la función **buscarPosició()** recursivamente hasta que la posición a la que llega es NULL donde se inserta.

4. Cada Node esta compuesto de: WordInfo más un puntero a su hijo izquierdo y derecho.

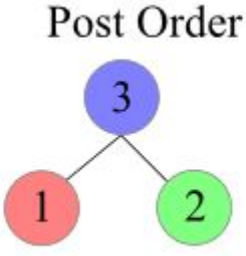
5. Al encontrar la posición correcta, se llama a la función **createNode()**, la cual reserva la memoria y copia los datos al node y luego le enlazamos al 'padre', dependiendo si tiene un valor mayor (right) o menor (left).

6. Buscar palabra dentro del árbol. Llamamos a la función **find_in_tree()**, que llama recursivamente **findNode()** que lo devuelve cuando coinciden las dos palabras. Si devuelve NULL es que la palabra no está en el diccionario.

7. Para imprimir el diccionario en orden alfabético, utilizamos el método **printTree()** que llama recursivamente a **printInOrder()**.



8. Para limpiar el árbol, utilizamos la función **clean_tree()**, que llama recursivamente **clearPostOrder()** con el node 'root'.



La función recursiva, **clearPostOrder(Node* node)**, se llama con el node→right, luego con el node→left i finalmente pone a null el node.
Hay una comprobación previa de si el node que recibe es NULL, no hace nada.

Estructuras

La estructura WordInfo es idéntica a la de la pràctica anterior, no hemos cambiado nada.

```
typedef struct {
    char word[MAX_WORD_LENGTH];
    char definition[MAX_DEFINITION_LENGTH];
    char pos;
} WordInfo;
```

La estructura Node tiene un puntero a su nodo izquierdo y derecho, ya que el Tree está compuesto de Nodes y al ser binario sólo puede tener dos hijos como màximo. La estructura Tree, tiene un entero que indica su tamaño y un puntero a la raíz del árbol (root).

```
typedef struct _Node {
    WordInfo data;
    struct _Node* right;
    struct _Node* left;
} Node;

typedef struct {
    Node* root;
    int size;
} Tree;
```

Las explicaciones de cada función están en el código de la práctica, explicadas en profundidad cada una.