Reto 4: Árboles

Francisco David Charte Luque Ignacio Cordón Castillo

1 Inorden no recursivo

Diseñar un procedimiento inorden no recursivo a imagen y semejanza del procedimiento preorden no recursivo que el profesor diseñó en la clase.

```
void inordenNR(const ArbolBinario<int>& a){
  ArbolBinario<int>::Nodo actual;
  stack<ArbolBinario<int>::Nodo> p;
  bool subiendo = false;

  actual=a.raiz();
  p.push(ArbolBinario<int>::nodo_nulo);
  p.push(actual);

  while (actual != ArbolBinario<int>::nodo_nulo){
      if (a.izquierda(actual) != ArbolBinario<int>::nodo_nulo && !subiendo){
          // Pasamos a manejar el hijo izquierdo
          actual = a.izquierda(actual);
          p.push(actual);
      } else {
          cout << a.etiqueta(actual) << ' ';
          p.pop();
          subiendo = true;

          if (a.derecha(actual) != ArbolBinario<int>::nodo_nulo) {
              // Pasamos a manejar el hijo derecho
              actual = a.derecha(actual);
              p.push(actual);
              subiendo = false;
          } else {
              // Trataremos de saltar al hermano
              actual = p.top();
          }
      }
```

```
    }
}
```

2 Codificación de árbol binario

Dar un procedimiento para guardar un árbol binario en disco de forma que se recupere la estructura jerárquica de forma unívoca usando el mínimo número de centinelas que veais posible.

Se propone lo siguiente:

Deseamos conocer para cada nodo del árbol si tiene dos hijos, sólo el izquierdo, solo el derecho o ninguno. Se empleará el preorden del árbol binario, haciendo que a cada nodo le preceda, caso de ser necesario, uno de los tres centinelas siguientes:

i. `<` Si le falta el hijo izquierdo

ii. `>` Si le falta el hijo derecho

iii. `-` Si no tiene hijos

No se hará empleo de ningún centinela si el nodo tiene ambos hijos.