Guía Práctica N° 4 – Adicional 2021
Árboles Binarios

Referencias:

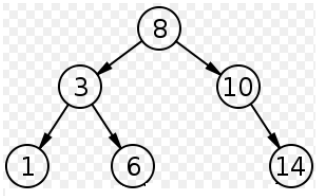

Figura 3.1

```
Arbol:
Nodo: 8 | Iz-> 3  De-> 10
        Nodo: 3 | Iz-> 1  De-> 6
                Nodo: 1 | Iz-> NULL  De-> NULL
                Nodo: 6 | Iz-> NULL  De-> NULL
        Nodo: 10 | Iz-> NULL  De-> 14
                Nodo: 14 | Iz-> NULL  De-> NULL
```

Figura 3.2

Recomendaciones:

- Utilizar funciones para el desarrollo.
- Probar los programas, en la máquina o con pruebas de escritorio.
- Utilizar nombres de fácil lectura para las variables.

3. Bis. Dado el árbol de la figura 3.1 siguiente:

1. Realice una estructura para la implementación.
Solución

```
struct nodo_arbol_binario {
        int dato;
        struct nodo_arbol_binario* iz = NULL;
        struct nodo_arbol_binario* de = NULL;
};
typedef struct nodo_arbol_binario NABinario;
```

2. Implemente forma "`hardcodeada`" de carga del árbol de la fig 3.1.
Solución

```
void generaArbol_2021_ej3_3bis(NABinario * &arbol) {
        NABinario *nuevo;
        NABinario *aux;

        nuevo = new (NABinario); //crea el nodo
        nuevo->dato = 8;  nuevo->iz = NULL; nuevo->de = NULL;
        arbol = nuevo;    // arbol tiene solo la raiz;

        //descendencia 8
        nuevo = new (NABinario); //crea el nodo
        nuevo->dato = 3;  nuevo->iz = NULL; nuevo->de = NULL;
        arbol->iz = nuevo;

        nuevo = new (NABinario); //crea el nodo
        nuevo->dato = 10;  nuevo->iz = NULL; nuevo->de = NULL;
        arbol->de = nuevo;

        //descendencia 3
        nuevo = new (NABinario); //crea el nodo
        nuevo->dato = 1;  //nuevo->iz = NULL; nuevo->de = NULL;
        arbol->iz->iz = nuevo;
        nuevo = new (NABinario); //crea el nodo
        nuevo->dato = 6;  nuevo->iz = NULL; nuevo->de = NULL;
        aux = arbol->iz;   // alternativa
        aux->de = nuevo;

        //descendencia 10
        nuevo = new (NABinario); //crea el nodo
        nuevo->dato = 14; // nuevo->iz = NULL; nuevo->de = NULL;
        aux = arbol->de;   // alternativa
        aux->iz = NULL;
        aux->de = nuevo;
}
```

```
int main (void)
{
        NABinario* arbol = NULL;

        generaArbol_2021_ej3_3bis(arbol);

        mostrar_2021_ej4_3bis(arbol);
        . . .

}
```

3. Muestre el contenido con un algoritmo `Pre-Orden`, `Post-Orden`, `In-Orden` usando recursión.

```
void abinario_preorden_recursivo (NABinario* arbol){  // Barrido RID
       if (arbol != NULL){
              /* Tratamiento del nodo en el algoritmo */
              cout << arbol->dato << "  ";
              /*fin tratamiento nodo */

              abinario_preorden_recursivo (arbol->iz);
              abinario_preorden_recursivo (arbol->de);
        }
}

void abinario_posorden_recursivo (NABinario* arbol){  // Barrido IDR
       if (arbol != NULL) {
              abinario_posorden_recursivo (arbol->iz);
              abinario_posorden_recursivo (arbol->de);
              /* Tratamiento del nodo en el algoritmo */
              cout << arbol->dato << "  ";
              /*fin tratamiento nodo */
       }
}


void abinario_inorden_recursivo (NABinario* arbol) {   // IRD
       if (arbol != NULL){
              abinario_inorden_recursivo (arbol->iz);

              /* Tratamiento del nodo en el algoritmo */
              cout << arbol->dato << "  ";
              /*fin tratamiento nodo */

              abinario_inorden_recursivo (arbol->de);
       }
}
```

4. Realice una representación que muestre "`fig 3.2`".

```
void mostrar_2021_ej4_3bis(NABinario* arbol, int tabulado) {
       if (arbol != NULL)    {
              cout << string (tabulado, '\t');
              cout << "Nodo: " << arbol->dato << " | " << "Iz-> ";
              if (arbol->iz != NULL)
                    cout << arbol->iz->dato;
              else
                    cout << "NULL";
              cout << "  " << "De-> ";
              if (arbol->de != NULL)
                     cout << arbol->de->dato;
              else
                     cout << "NULL";
              cout << endl;

              tabulado++;

              abinario_mostrar_recursivo (arbol->iz, tabulado);
              abinario_mostrar_recursivo (arbol->de, tabulado);
       }
}
```

5. Realice una función que cuente la cantidad de nodos del árbol. Queda como tarea.