LABORATORIO 5 – GUIA 1
CURSO ESTRUCTURAS DISCRETAS II

Apellidos y Nombres: .....
CUI: ..... email (Institucional): .....
Fecha: 05 de octubre 2020 Sección: A y B
Docente: Dra. Roxana Flores Quispe /Dr. Yuber Elmer Velazco Paredes

Guía N° 1 : Arboles Binarios de Búsqueda

Alumno: _____	**Nota:**

I. Objetivos

- Presentar al alumno la implementación de arboles binarios de búsqueda en C++.

II. Equipos y materiales

Equipos y dispositivos:

- PC.

Software:

- C++.

III. Actividades

a) Estructuras y tipos de datos definidos (sinónimos).
Las **estructuras** nos permiten agrupar varios datos, que mantienen algún tipo de relación, aunque sean de distinto tipo, permitiendo manipularlos todos juntos usando un mismo identificador, o cada uno por separado.
Un **typedef** es una palabra reservada en el lenguaje de programación C y C++. Su función es asignar un nombre alternativo a tipos existentes.

```
#include <ctime>
#include <cstdlib>
#include <iostream>
using namespace std;

typedef int entero;
// typedef usado para definir sus propios tipos a partir de los tipos base.

struct coord{
   int x, y;
```

```
};

coord inicializa_punto1()

{
   coord t0;
   t0.x = rand()%1024;
   t0.y = rand()%768;
   return t0;
}

coord inicializa_punto2(int x, int y)
{
   coord t0;
   t0.x = x;
   t0.y = y;
   return t0;
}

int main()
{
   coord pt1, pt2;
     srand((unsigned) time(0)); // cambiar la semilla

   pt1 = inicializa_punto1();
   pt2 = inicializa_punto2(40, 40);
     cout << " pt1.x=" << pt1.x << " pt1.y=" << pt1.y << endl;

     entero x,y;
     x = pt2.x;
     y = pt2.y;
     cout << " pt2.x=" << x << " pt2.y=" << y << endl;


   return 0;
}
```

b) Arboles binarios de búsqueda y sus recorridos.

Un árbol **binario de búsqueda** o ABB, es un árbol binario en el cual para todo elemento, los elementos mayores a él, se ubican en su rama derecha, mientras que los elementos menores van en su rama izquierda. Cada elemento se almacena una sola vez por lo que no existen elementos repetidos.

Existen varias formas de realizar el **recorridos de una árbol,** sin embargo existes tres formas básicas y son:

Enorden: Primero el hijo izquierdo, luego el padre y finalmente el hijo derecho
Preorden: Primero el padre, luego el hijo izquierdo y finalmente el hijo derecho.
Postorden: Primero hijo izquierdo, luego el hijo derecho y finalmente el padre

```
#include <iostream>
using namespace std;

struct nodo{
      int nro;
   struct nodo *izq, *der;
};

typedef struct nodo *ArbolBinario;

ArbolBinario crearNodo(int x)
{
   ArbolBinario nuevoNodo = new(struct nodo);
   nuevoNodo->nro = x;
   nuevoNodo->izq = NULL;
   nuevoNodo->der = NULL;

   return nuevoNodo;
}

void insertar(ArbolBinario &arbol, int x)
{
   if(arbol==NULL)
      arbol = crearNodo(x);
   else if(x < arbol->nro)
      insertar(arbol->izq, x);
   else if(x > arbol->nro)
      insertar(arbol->der, x);
}

void preOrden(ArbolBinario arbol)
{
   if(arbol!=NULL)
   {
      cout << arbol->nro <<" ";
      preOrden(arbol->izq);
      preOrden(arbol->der);
     }
}
```

```
void enOrden(ArbolBinario arbol)
{
   if(arbol!=NULL)
   {
      enOrden(arbol->izq);
      cout << arbol->nro << " ";
      enOrden(arbol->der);
   }
}

void postOrden(ArbolBinario arbol)
{
   if(arbol!=NULL)
   {
      postOrden(arbol->izq);
      postOrden(arbol->der);
      cout << arbol->nro << " ";
   }
}

int main()
{
   ArbolBinario arbol = NULL;

   cout << "\n ...EJEMPLO DE ARBOL BINARIO...  \n";

   insertar(arbol, 4);
     insertar(arbol, 8);
     insertar(arbol, 2);
     insertar(arbol, 9);
     insertar(arbol, 3);
     insertar(arbol, 6);
     insertar(arbol, 1);
     insertar(arbol, 7);
     insertar(arbol, 5);

   cout << "\nRecorridos del ArbolBinario\n";
   cout << "\nEn orden   :  ";   enOrden(arbol);
   cout << "\nPre Orden  :  ";   preOrden(arbol);
   cout << "\nPost Orden :  ";   postOrden(arbol);
   cout << endl << endl;

   return 0;
```

```
}
```

IV. Resultados

El alumno comprende el tipo de dato árbol binario de búsqueda y sus recorridos.