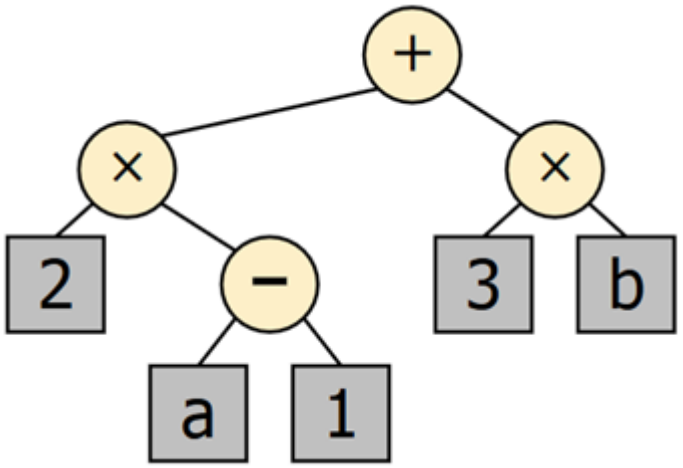
1. Defina con sus palabras qué es un Árbol de datos.

En ciencias de la computación y en informática, un árbol es un tipo abstracto de datos (TAD) ampliamente usado que imita la estructura jerárquica de un árbol, con un valor en la raíz y subárboles con un nodo padre, representado como un conjunto de nodos enlazados.

2. Cree la siguiente estructura de datos en un programa C. Luego ingrese los valores indicados en los nodos del árbol tal como lo indica la figura.




```
#include <stdio.h>

#include <malloc.h>

#include <stdlib.h>


struct NODO {

        char dato;

        struct NODO *izq;

        struct NODO *der;

};
```

```
struct NODO *nuevoNodo (int item);
struct NODO*insertar (struct NODO* NODO, int item);
void imprimirINORDER (struct NODO *NODO);

int main (){
	struct NODO *raiz = NULL;
	raiz = insertar(raiz, '+');
	insertar(raiz, 'x');
	insertar(raiz, '2');
	insertar(raiz, '-');
	insertar(raiz, 'a');
	insertar(raiz, '1');
	insertar(raiz, '*');
	insertar(raiz, '3');
	insertar(raiz, 'b');


	imprimirINORDER(raiz);

}

struct NODO *nuevoNodo (int item){
	struct NODO *temporal = (NODO *)malloc(sizeof(NODO));
	temporal->dato = item;
	temporal->der = NULL;
```

```
	temporal->izq = NULL;
	return temporal;
}
struct NODO *insertar (struct NODO* NODO, int item){
	if (NODO == NULL){
		return nuevoNodo(item);
	}

	if (item < NODO->dato){
		NODO->izq = insertar(NODO->izq, item);
	} else if ( item > NODO->dato){
		NODO->der = insertar(NODO->der, item);
	}

	return NODO;
}
void imprimirINORDER (struct NODO *NODO){
	if(NODO == NULL){
		return;
	}
	imprimirINORDER(NODO->izq);
	printf ("%c ", NODO->dato);
	imprimirINORDER(NODO->der);
}
```

3. Programa en C. Implemente una función que devuelva la amplitud de un árbol completo y balanceado.

4. Programa en C. Implemente una función que devuelva el alto de un árbol binario.

```
#include <stdio.h>
#include <malloc.h>

struct Nodo{

	int dato;
	Nodo *izq;
	Nodo *der;
};

struct Nodo *crearNodo(int dato);
struct Nodo *insertar(struct Nodo *nodo, int dato);
void imprimeInorder(struct Nodo *nodo);
int amplitudArbol(struct Nodo *nodo, int *a);
int alturaArbol(struct Nodo *nodo);
int max(int a, int b);

main(){

	Nodo *raiz = NULL;
```

```
        int a = 0;

        raiz = insertar(raiz, 50);
    insertar(raiz, 40);
    insertar(raiz, 20);
    insertar(raiz, 45);
    insertar(raiz, 60);
    insertar(raiz, 55);
    insertar(raiz, 70);

    printf ("El arbol es: \n");
    imprimeInorder(raiz);

    printf ("\n\nLa amplitud del arbol es: %d", amplitudArbol(raiz, &a));
    printf ("\n\nLa altura del arbol es: %d", alturaArbol(raiz));

}

struct Nodo *crearNodo(int dato){

        Nodo *nuevo = (Nodo *) malloc (sizeof(Nodo));

        nuevo->dato = dato;
        nuevo->izq = NULL;
        nuevo->der = NULL;
```

```
        return nuevo;
}

struct Nodo *insertar(struct Nodo *nodo, int dato){

    if (nodo == NULL){

                return crearNodo(dato);
        }

    if (dato < nodo->dato){

        nodo->izq = insertar(nodo->izq, dato);
        }
    else if (dato > nodo->dato){

        nodo->der = insertar(nodo->der, dato);
        }

    return nodo;
}

void imprimeInorder(struct Nodo *nodo){

        if (nodo == NULL){
```

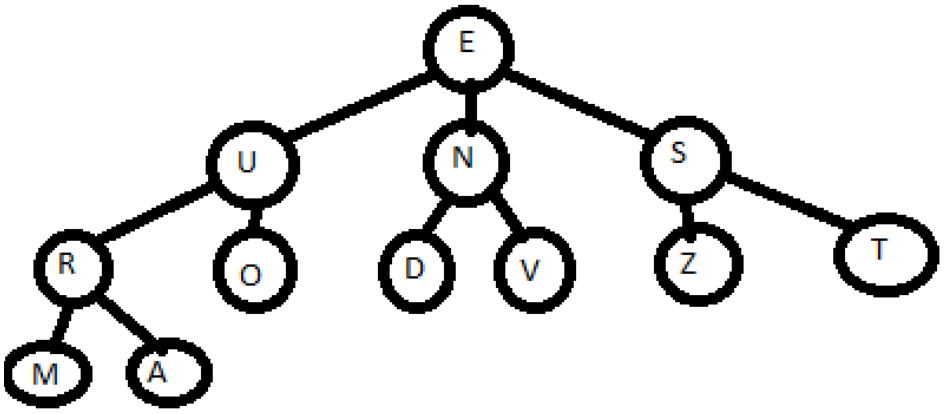
```
                return;
        }

    imprimeInorder(nodo->izq);
    printf("%d ", nodo->dato);
    imprimeInorder(nodo->der);
}

int amplitudArbol(struct Nodo *nodo, int *a){

        if (nodo == NULL || nodo->izq == NULL && nodo->der == NULL){

                *a = *a + 1;
                return *a;
        }

    amplitudArbol(nodo->izq, a);
    amplitudArbol(nodo->der, a);
    return *a;
}

int alturaArbol(struct Nodo *nodo){

        if (nodo == NULL){
```

```
            return 0;
        }

    return 1 + max(alturaArbol(nodo->izq),alturaArbol(nodo->der));
}

int max(int a, int b){

        if (a>b){

            return a;
        }
        else{

           return b;
        }
}
```

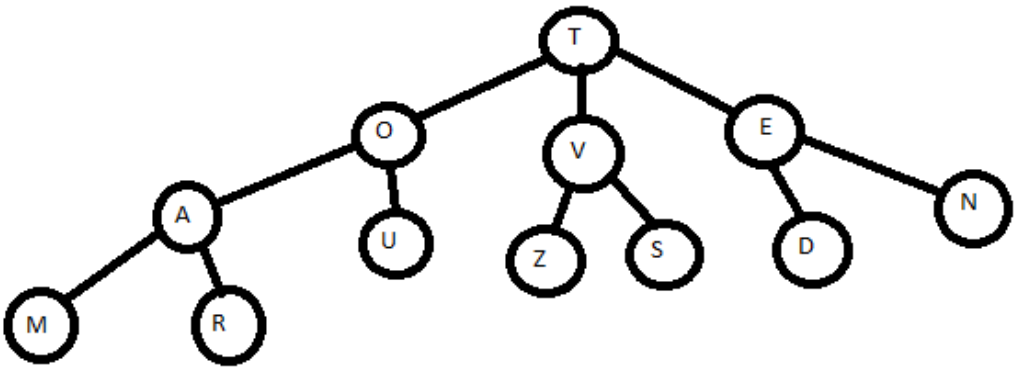
5. Programa en C. Escribir una función recursiva que encuentre el número de nodos de un árbol binario.

6. Determinar la estructura de un Árbol cuyo recorrido pre order es E,U,R,M,A,O,N,D,V,S,Z,T, e in order es M,R,A,U,O,Z,S,V,D,N,E,T. Indicar el resultado de su recorrido post orden.

PRE-ORDER:



IN ORDER:



7. Programa en C. Implemente las funciones preorder, inorder y postorder para recorrer un árbol binario completo.

8. Programa en C. Escribir una función que invierta el árbol binario pasado por parámetro, de manera tal que la rama izquierda de la raíz se convierta en la rama derecha.c

9. Programa en C. Escribir un árbol con números enteros como datos. Luego reemplazar cada nodo por el resultado de la multiplicación de sus dos hijos. Imprimir el resultado de dicho procesamiento con un preorder, inorder o postorder.

10. Programa en C. Implemente una función que tome los elementos de un árbol binario completo y los transfiera a un arreglo.