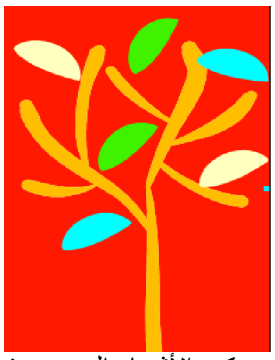
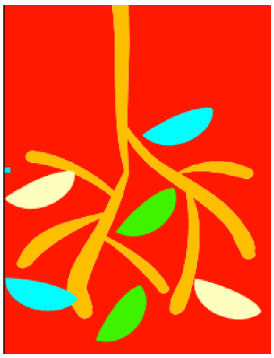
الأشجار
Trees

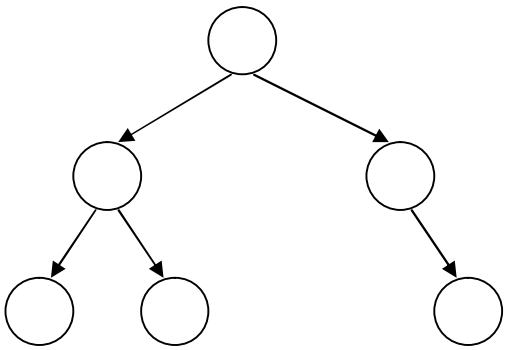
1- تعريف

الشجرة مجموعة من العناصر المترابطة حسب تسلسل هرمي.



في علم الحاسب، يقع تصوير الأشجار على عكس الأشجار الموجودة في الطبيعة (جذر رئيسي تتفرع عنه مجموعة من الأغصان والأوراق)



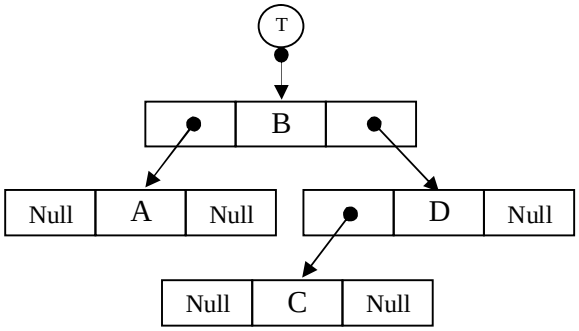


في هذا الدرس سيقع الاهتمام بنوع خاص من الأشجار وهو الأشجار الثنائية (Binary Trees) حيث يتفرع عن كل غصن رئيسي غصنان ثانويان: أيمن و أيسر.

2- تمثيل الأشجار الثنائية

تعتبر الأشجار نوع خاص من القوائم المتصلة حيث يحتوي كل عنصر على مؤشرين:

- الأول يشير إلى الشجرة الثانوية اليسرى (Left Subtree)
- الثاني يشير إلى الشجرة الثانوية اليمنى (Right Subtree)



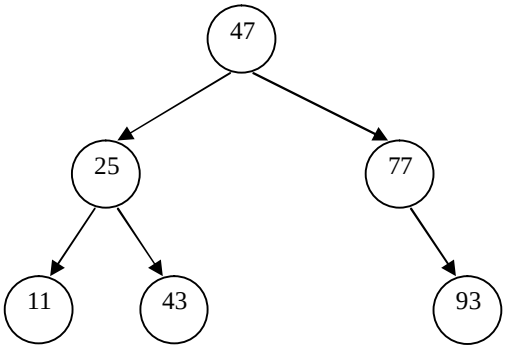
- يسمى العنصر الأول في الشجرة (B) **جذر الشجرة** (The root node)
- يشير كل سهم إلى أحد **الأطفال** (a child)
- يسمى العنصر (A) جذر **الشجرة الثانوية** اليسرى (The root of the left subtree)
- يسمى العنصر (D) جذر الشجرة الثانوية اليمنى (The root of the right subtree)
- يسمى أطفال العنصر الواحد **أشقاء** (Siblings) (مثل العقد A و D)
- يسمى العنصر الذي لا أطفال له **ورقة** (Leaf node) (مثل العقد A و C).

3- أشجار البحث الثنائية

يتميز هذا النوع من الأشجار بالخصائص التالية:

- عدم وجود أي تشابه بين العناصر
- كل عناصر الشجرة الثانوية اليسرى أقل من العنصر الأب
- كل عناصر الشجرة الثانوية اليمنى أكبر من العنصر الأب

مثال



في لغة C++ يمكن الإعلان عن الأشجار الثنائية كما يلي:

```
struct TreeNode
{
        TreeNode  *LeftPtr;
        int item;
        TreeNode  *RightPtr;
};

TreeNode  *T;
```

تتم معالجة عناصر الأشجار الثنائية المرتبة من خلال الدوال التالية:

الدالة	دورها
Tree* InsertNode(int x, TreeNode *T)	إضافة العنصر x إلى الشجرة T في (المكان المناسب)
void PreOrderTraversal(TreeNode *T)	اجتياز وطباعة العناصر حسب طريقة الجذر أولا (Pre Order)
void InOrderTraversal(TreeNode *T)	اجتياز وطباعة العناصر حسب طريقة اليسار أولا (In Order)
void PostOrderTraversal(TreeNode *T)	اجتياز وطباعة العناصر حسب طريقة اليمين أولا (Post Order)

يقوم البرنامج التالي بإنشاء شجرة ثنائية ثم اجتيازها وطباعة عناصرها على الشاشة حسب طريقة in order، Pre order و Post order.

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

struct TreeNode
{
        TreeNode *LeftPtr;
        int Item;
        TreeNode *RightPtr;
};
TreeNode* InsertNode(int x, TreeNode *T)
{
        if (T == NULL)
        {
              TreeNode *R = new(TreeNode);
              R->Item = x;
              R->LeftPtr = NULL;
              R->RightPtr = NULL;
              T = R;
        }
        else
        {
              if (x < T -> Item)
                    T -> LeftPtr = InsertNode(x, T -> LeftPtr);
              else
                    T -> RightPtr = InsertNode(x, T -> RightPtr);
        }
        return T;
}
void PreOrderTraversal(TreeNode *T)
{
        if (T != NULL)
        {
              cout<<T -> Item<<"\t";
              PreOrderTraversal(T->LeftPtr);
              PreOrderTraversal(T->RightPtr);
        }
}
```

```
void InOrderTraversal(TreeNode  *T)
{
	if (T != NULL)
	{
		InOrderTraversal(T->LeftPtr);
		cout<<T -> Item<<"\t";
		InOrderTraversal(T->RightPtr);
	}
}
void PostOrderTraversal(TreeNode  *T)
{
	if (T != NULL)
	{
		PostOrderTraversal(T->LeftPtr);
		PostOrderTraversal(T->RightPtr);
		cout<<T -> Item<<"\t";
	}

}
void main()
{
	TreeNode *T = NULL;

	int x;
	cout<<"Enter an item (or 0 for end) ):"; cin>>x;
	while (x != 0)
	{
		T = InsertNode(x, T);
		cout<<"Enter an item (or 0 for end) ):"; cin>>x;
	}
	PreOrderTraversal(T);
	cout<<endl;
	InOrderTraversal(T);
	cout<<endl;
	PostOrderTraversal(T);
	cout<<endl;
}
```

المخرجات على الشاشة

```
Enter an item (or 0 for end) ):50
Enter an item (or 0 for end) ):25
Enter an item (or 0 for end) ):75
Enter an item (or 0 for end) ):100
Enter an item (or 0 for end) ):200
Enter an item (or 0 for end) ):0
Pre-order Traversal: 50 25      75      100     200
In-order traversal: 25  50      75      100     200
Post-order traversal: 25        200     100     75      50
Press any key to continue_
```

تمرين: أكمل البرنامج التالي الذي من المفترض أن يقوم بالتعريف بالتصنيف TreeNode

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

…… TreeNode
{
        private:
                TreeNode  *LeftPtr;
                int item;
                ……  *RightPtr;

        ……
                TreeNode* InsertNode(int);
                ……
                void InOrderTraversal(void);
                ……
};

TreeNode* ……::InsertNode(int  x)
{
        TreeNode *T = this;
        if (T == NULL)
        {
                TreeNode  *R = new(TreeNode);
                R -> item  = x;
                R -> LeftPtr = ……;
                R -> RightPtr = ……;
                T = R;
        }
        else
        {
                if (……)
                        T->LeftPtr = (T->LeftPtr)->InsertNode(x);
                else
                        T->RightPtr = (T->RightPtr)->InsertNode(x);
        }
        return T;
}

void TreeNode::PreOrderTraversal(void)
{
        if (this != NULL)
        {
                cout<<this->item<<"\t";
                (this->LeftPtr)->PreOrderTraversal();
                ……
        }
}
 void TreeNode::InOrderTraversal(void)
{
```

```
        if (this != NULL)
        {
                (this->LeftPtr)->InOrderTraversal();
                cout<<this->item<<"\t";
                (this->RightPtr) -> InOrderTraversal();
        }
}

void TreeNode::PostOrderTraversal(void)
{
        if (this != NULL)
        {
                ……
                ……
                ……..
        }
}

void main()
{

        TreeNode *T = NULL;
        int x;
        cout<<"Enter an element (or 0 for end) :";
        cin>>x;
        while (……)
        {
                T = ……
                cout<<"Enter an element (or 0 for end) :";
                cin>>x;
        }

        cout<<"Pre-order Traversal:"; T->PreOrderTraversal();
        cout<<endl;
        cout<<"In-order Traversal:"; T->InOrderTraversal();
        cout<<endl;
        cout<<"Post-order Traversal:"; T->PostOrderTraversal();
        cout<<endl;
}
```

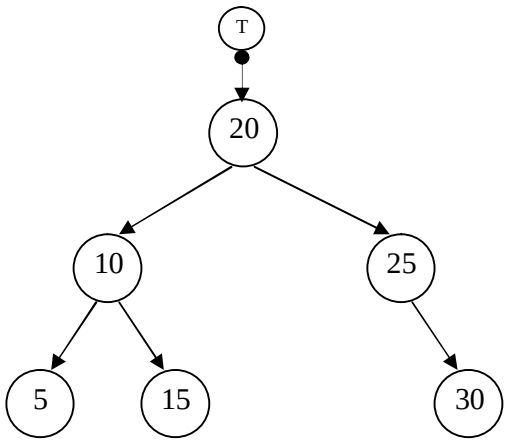
تمرين

أكتب برنامجا بلغة C++ يقوم بما يلي:

1- إنشاء المصفوفة A وتعبئتها بستة عناصر يقوم المستخدم بإدخالها عن طريق لوحة المفاتيح
2- تحويل المصفوفة A إلى شجرة بحث ثنائية T
3- طباعة عناصر الشجرة على الشاشة بشكل مرتب

مثال

A	20	25	5	15	30	10



- **البرنامج**

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

struct TreeNode
{
        TreeNode *LeftPtr;
        int item;
        TreeNode *RightPtr;
};

 TreeNode*  InsertNode(int  x, TreeNode  *T)
{
        if (T == NULL)
        {
                TreeNode  *R = new(TreeNode);
                R -> item  = x;
                R -> LeftPtr = NULL;
                R -> RightPtr = NULL;
                T = R;
        }

        else
```

```
        {
                if (x < T->item)
                        T->LeftPtr = InsertNode(x, T->LeftPtr);
                else
                        T->RightPtr = InsertNode(x, T->RightPtr);
        }
        return T;
}
void InOrderTraversal(TreeNode  *T)
{
        if (T != NULL)
        {
                InOrderTraversal(T->LeftPtr);
                cout<<T->item<<"\t";
                InOrderTraversal(T->RightPtr);
        }
}

void main()
{

        int A[10];

        TreeNode *T = NULL;

        int i;

        for (i = 0; i<= 9; i++)
        {
                cout<<"Enter an integer :";
                cin>>A[i];
        }

        for (i = 0; i<= 9; i++)
        {
                T = InsertNode(A[i],T);
        }

        cout<<"In-order Traversal:";
        InOrderTraversal(T);
        cout<<endl;
}
```