



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

*Институт информационных технологий
Кафедры Вычислительной техники*

Лабораторная работа № 1 Вариант 4, 18

По дисциплине «Структуры и алгоритмы обработки данных»

Выполнил студент группы ИКБО-13-18, Конде Дауда
(учебная группа, Фамилия Имя Отчество студента).
студента)


(подпись)

Преподаватель	<u>ассистент, Расулов М.М.</u>
_____	_____
	Должность, звание, ученая степень
	подпись

Работа представлена к защите «___» _____ 2020 г.

Оценка «_____»

Задачи:

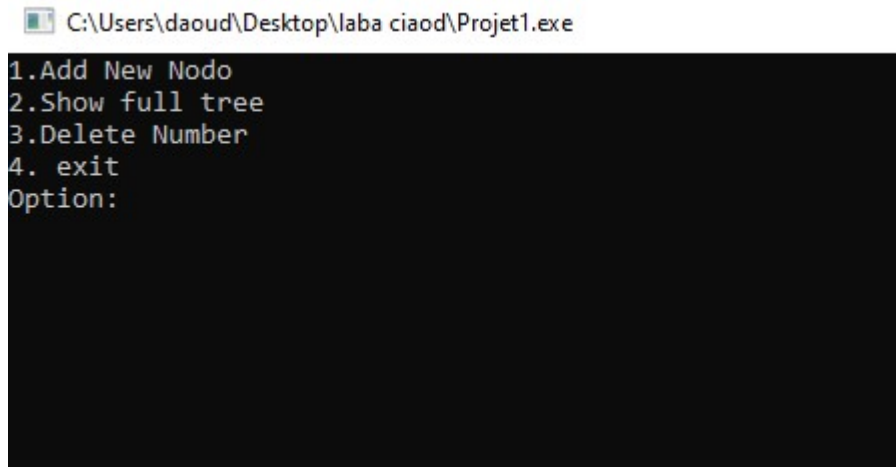
Вариант 4 :

Текстовая визуализация дерева (значение каждого узла выводится на отдельной строке, с отступом пропорциональным глубине узла (шаг отступа равен двум пробелам), в порядке старшинства узлов).

Вариант 18 :

Удалите все вершины, у которых высота левого поддерева отличается от высоты правого поддерева на 2.

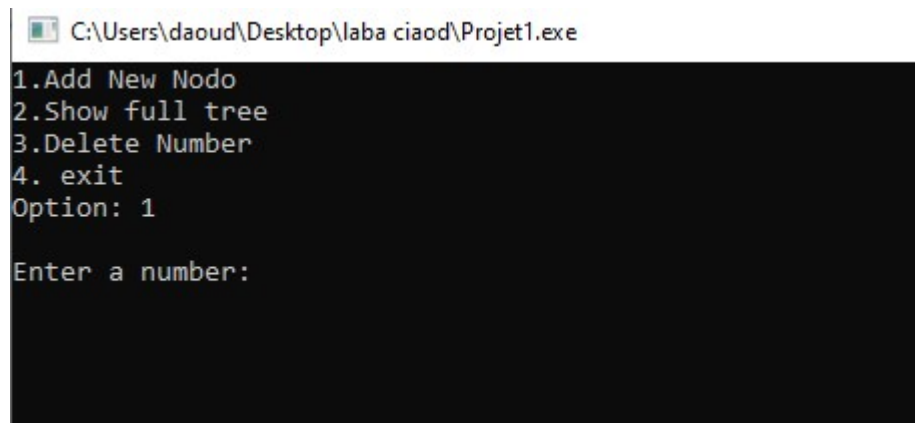
Чтобы отобразить различные элементы, мы должны сначала ввести элементы.



```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Nodo
2.Show full tree
3.Delete Number
4. exit
Option:
```

Рисунок 1

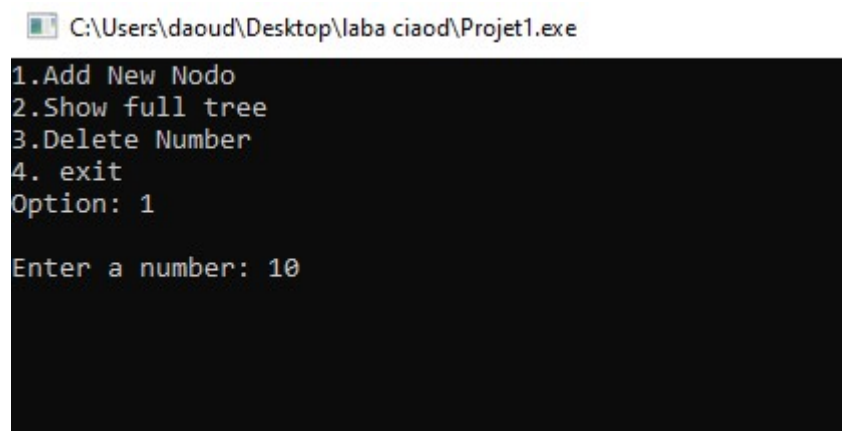
к каждой записи 1 мы добавляем элементы, 2, что позволяет нам отображать элементы, 3 позволяет нам удалять элементы и 4, чтобы выйти из программы.



```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Nodo
2.Show full tree
3.Delete Number
4. exit
Option: 1
Enter a number:
```

Рисунок 2

при каждой записи 1 мы должны добавлять элементы для заполнения.



```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Nodo
2.Show full tree
3.Delete Number
4. exit
Option: 1
Enter a number: 10
```

Рисунок 3

добавьте 10, следующие цифры будут добавлены в программу, слева мы имеем, 5, 3, 8, 6, 9 и 7, а справа от дерева, мы имеем 15, 12, 20, 30

```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Nodo
2.Show full tree
3.Delete Number
4. exit
Option: 2
```

Рисунок 4

После ввода элементов мы отображаем элементы с опцией 2.

```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Nodo
2.Show full tree
3.Delete Number
4. exit
Option: 2
Show full tree:
      30
     20
    15
   12
  10
   9
  8
   7
  6
 5
 3
Appuyez sur une touche pour continuer...
```

Рисунок 5

Для удаления предметов, мы проходим следующую процедуру.

Вариант 4 для удаления элементов,

```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Node
2.Show full tree
4.Delete Number
5. exit
Option:4
```

Рисунок 6

```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Node
2.Show full tree
4.Delete Number
5. exit
Option:4

ENTER NUMBER TO DELETE:10

Appuyez sur une touche pour continuer...
```

Рисунок 7

удаление 10

```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Node
2.Show full tree
4.Delete Number
5. exit
Option:2

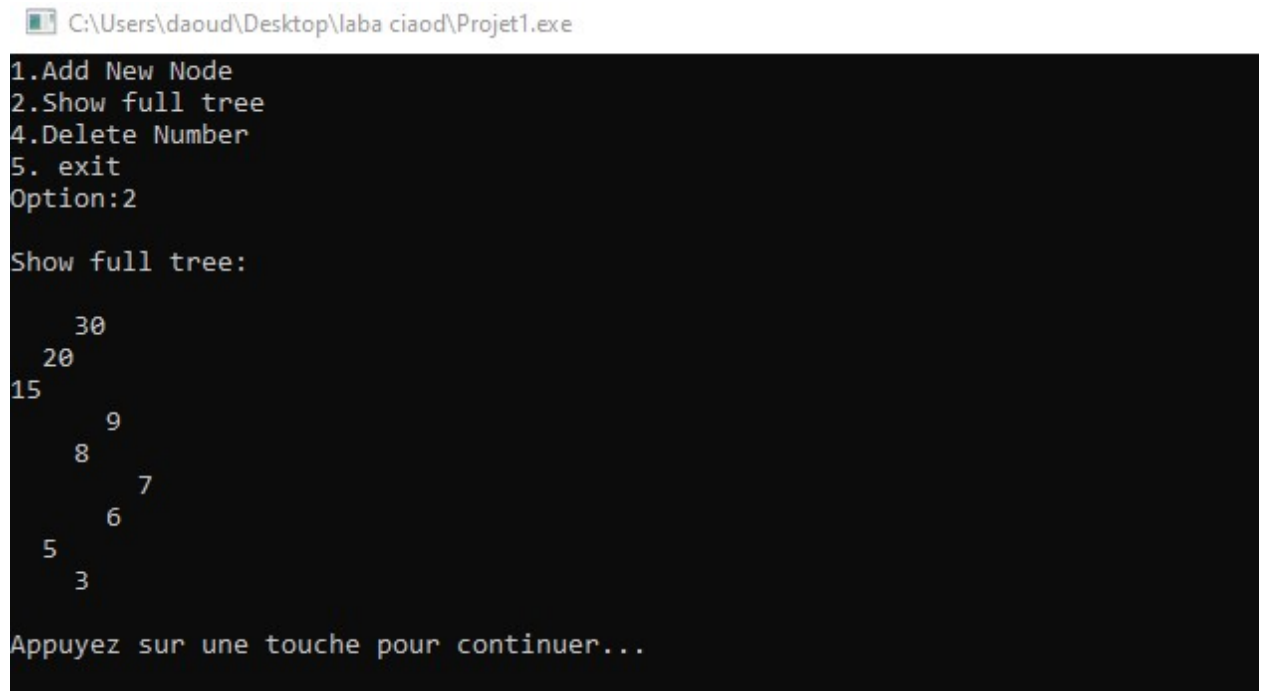
Show full tree:

    30
   20
  15
12
   9
   8
    7
   6
   5
   3

Appuyez sur une touche pour continuer...
```

Рисунок 7

удаление 12



```
C:\Users\daoud\Desktop\laba ciaod\Projet1.exe
1.Add New Node
2.Show full tree
4.Delete Number
5. exit
Option:2
Show full tree:
    30
   20  15
  9    6
 8    5
7    3
Appuyez sur une touche pour continuer...
```

Рисунок 7

ИСХОДНЫЙ КОД :

```
#include <iostream>
#include <stdlib.h>

using namespace std;

struct Nodo{
int dato;
Nodo *right;
Nodo *left;
Nodo *padre;
};
//the prototype of function
void menu();
Nodo *crearNodo(int, Nodo *);
void AddNodo(Nodo *&, int, Nodo *);
void ShowTree(Nodo *, int);
void deleteA(Nodo *,int);
void deleteNodo(Nodo *);
Nodo *minimo(Nodo *);
void PosOrden(Nodo *);
void reemplazar(Nodo *,Nodo *);
void DestruirNodo(Nodo *);

Nodo *arbol= NULL;
```

```

int main()
{
    menu();

    return 0;
}
//funcion de menu
void menu(){
    int dato, option, cont=0;
    do{
        cout<<"1.Add New Node"<<endl;
        cout<<"2.Show full tree"<<endl;
        cout<<"4.Delete Number"<<endl;
        cout<<"5. exit"<<endl;
        cout<<"Option:";
        cin>>option;
        switch(option){
            case 1: cout<<"\nEnter a number: ";
                cin>>dato;
                AddNodo(arbol,dato,NULL);//insertion of elements
                cout<<"\n";
                break;
            case 2: cout<<"\nShow full tree:\n\n";
                ShowTree(arbol, cont);
                cout<<"\n";
                system("pause");
                break;
            PosOrden(arbol);
                cout<<"\n\n";
                system("pause");
                break;
            case 4 :cout<<"\nENTER NUMBER TO DELETE:";
                cin>>dato;
                deleteA(arbol, dato);
                cout<<"\n";
                system("pause");
                break;
        }
        system("cls");
    }while(option != 5);

}
//funcion para crear un nuevo nodo
Nodo *crearNodo(int n, Nodo *padre){
    Nodo *new_nodo = new Nodo();
    new_nodo->dato = n;
    new_nodo->right=NULL;

```

```

    new_nodo->left=NULL;
    new_nodo->padre=padre;

    return new_nodo;
}
// функция для добавления элементов в дерево
void AddNodo(Nodo *&arbol, int n, Nodo *padre){
    if(arbol == NULL){
        Nodo*new_nodo=crearNodo(n,padre);
        arbol =new_nodo;
    }
    else{// если дерево имеет новый узел или более
        int valorRaiz= arbol->dato;// мы получаем значение корня
        if(n < valorRaiz){// если элемент меньше, чем корень, мы добавляем слева
            AddNodo(arbol->left, n,arbol);
        }
        else{// если элемент больше, мы добавляем справа
            AddNodo(arbol->right,n,arbol);
        }
    }
}
// функция для отображения дерева
void ShowTree(Nodo *arbol, int cont){
    if (arbol ==NULL){
        return;
    }
    else{
        ShowTree(arbol->right,cont+1);
        for(int i=0; i<cont;i++)
            cout<< " ";
    }
    cout<<arbol->dato<<endl;
    ShowTree(arbol->left, cont+1);
}

void deleteA(Nodo *arbol, int n){
    if(arbol== NULL){//si el arbol esta vacio
        return;// no hace nada
    }
    else if(n < arbol->dato){
        deleteA(arbol->left,n);
    }

    else if(n > arbol->dato){
        deleteA(arbol->right,n);
    }
    else{
        deleteNodo(arbol);
    }
}

```



```

}
}
// функция для определения самого левого узла
Nodo *minimo(Nodo *arbol){
    if(arbol == NULL){//si el arbol esta vacio
        return NULL;//retorna nulo
    }
    if(arbol->left){
        return minimo(arbol->left);
    }
    else{
        return arbol;
    }
}
// функция для замены двух узлов
void reemplazar(Nodo *arbol, Nodo * new_nodo){
    if(arbol->padre){
        //arbol->padre hay que asignarle su nuevo hijo
        if(arbol->dato == arbol->padre->left->dato){
            arbol->padre->left=new_nodo;
        }
        else if(arbol->dato == arbol->padre->right->dato){
            arbol->padre->right = new_nodo;
        }
    }
    if(new_nodo){
        // мы приступаем к назначению его нового отца
        new_nodo->padre = arbol->padre;
    }
}
//the function

void DestruirNodo(Nodo *Nodo){
    Nodo->left=NULL;
    Nodo->right = NULL;
    delete Nodo;
}
void deleteNodo(Nodo *nodoDelete){
    if(nodoDelete->left && nodoDelete->right){
        Nodo *menor = minimo(nodoDelete->right);
        nodoDelete->dato = menor->dato;
        deleteNodo(menor);
    }

    else if(nodoDelete->left){// если у тебя есть левый сын
        reemplazar(nodoDelete,nodoDelete->left);
        DestruirNodo(nodoDelete);
    }
}

```

```

else if(nodoDelete->right){
    reemplazar(nodoDelete,nodoDelete->right);
    DestruirNodo(nodoDelete);
}
else{
    reemplazar(nodoDelete,NULL);
    DestruirNodo(nodoDelete);
}
}
}
// функция заказа
void PosOrden(Nodo *arbol){
    if(arbol == NULL){
        return;
    }
    else{
        PosOrden(arbol->left);
        PosOrden(arbol->right);
        cout<<arbol->dato<<" -";
    }
}
}

```

Вывод

Реализация этих лабораторий позволила мне получить знания о деревьях, их функционировании и обработке. Я знал, как добавить элементы в дерево и удалить элементы из дерева.