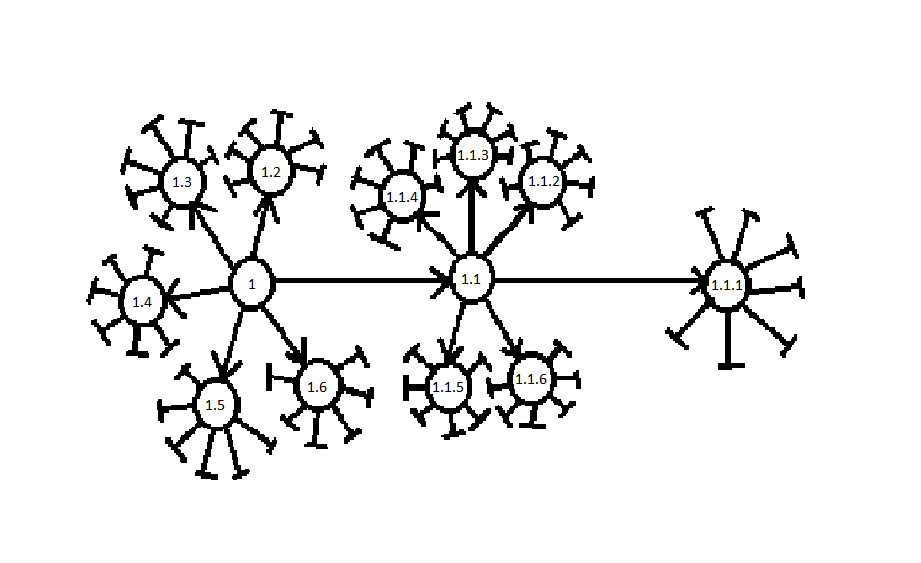
**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**  
Федеральное государственное бюджетное образовательное учреждение высшего образования  
**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА (МИИТ)»  
 РУТ (МИИТ)**  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Кафедра: «Вычислительные системы и сети»  
  
  
  
  
  
  
  
  
  
  
**О Т Ч Ё Т  
П О Л А Б О Р А Т О Р Н О Й Р А Б О Т Е № 6**  
по дисциплине «Структуры и алгоритмы обработки данных»  
на тему: «Сетевые структуры»  
  
  
  
  
  
  
  
  
  
 Выполнил: студент группы УИБ-311   
 Владыка П.А.

Москва 2018

**Задание**

 Разработать и отладить программу, реализующую сетевую структуру с заданной дисциплиной связей.

|  |  |
| --- | --- |
| № варианта | Вид сети |
| 4 | 1:6 |

Пример структуры, соответствующей дисциплине связей в варианте, которая может реализовать программа:

**Таблица имён**

|  |  |
| --- | --- |
| **Исходные данные** | |
| Knot | Указатель на первый элемент структуры (поле класса USESTRUCTURE) |
| **Промежуточные данные** | |
| Element | Указатель на первый элемент дерева (входной параметр функции) |
| Value | Имя переменной, хранящее вводимое с клавиатуры значение (поле класса USESTRUCTURE) |
| **Поля структуры** ELEMENT | |
| inf | Информационное поле |
| pointers | Массив указателей на элементы данного узла |

**Блок-схема**

**Главная программа**



**Исходный код**

/\* Лабораторная работа #6. Вариант-4. Дисциплина связей 1:6 \*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<iostream>

#include<windows.h>

#include <iomanip>

#include <malloc.h>

using namespace std;

/\* Класс создан для работы с бинарным деревом \*/

class NETST{

public:

template<typename ClassName, typename Type>

inline static bool AddElementToStructure(ClassName\*\* Element, Type value) {

if (\*Element != NULL) {

for (int i = 0; i < 6; i++) {

if ((\*Element)->Pointers[i] == NULL) {

(\*Element)->Pointers[i] = new ClassName;

(\*Element)->Pointers[i]->inf = value;

return true;}}

AddElementToStructure(&((\*Element)->Pointers[0]), value);}

else {

\*Element = new ClassName;

(\*Element)->inf = value;}

return true; }

template<typename ClassName>

inline static bool DeleteStructure(ClassName\*\* Element){

if (\*Element != NULL){

for (int i = 0; i < 6; i++) {

if ((\*Element)->Pointers[i] != NULL) {

DeleteStructure(&((\*Element)->Pointers[i]));

(\*Element)->Pointers[i] = NULL;}}

delete \*Element;

return true;}

return false;}};

/\* Класс для работы с сетевой структурой \*/

template<typename T>

class USETSTRUCTURE {

public:

USETSTRUCTURE() : Knot(NULL) {}

~USETSTRUCTURE() {NETST::DeleteStructure(&Knot);}

void menu() {

char choice;

char secondchoise;

system("cls");

cout << " {{>================================================<}}\n"

<< " || Сетевая структура ||\n"

<< " {{>================================================<}}\n"

<< " || 1. Добавить элемент в структуру ||\n"

<< " || 2. Просмотр элементов структуры (рекурсия) ||\n"

<< " || 3. Выход ||\n"

<< " {{>================================================<}}\n"

<< "\n Выбранная опция: ";

(cin >> choice).get();

switch (choice){

case 49:{

system("cls");

cout << "Введите новый элемент: ";

(cin >> Value).get();

AddElement(Value);

cout << "Добавлено. ";

system("pause");

menu();

break;}

case 50:{

system("cls");

if (!PrintTheStucture(&Knot))

cerr << "No structure";

cout << endl;

system("pause");

menu();

break;}

case 51:{break;}

default: menu();}}

private:/\* Структура элемента сетевой структуры \*/

template<typename Ty>

struct ELEMENT{

ELEMENT() {

for (int i = 0; i < 6; i++)

Pointers[i] = NULL;}

Ty inf;

ELEMENT\* Pointers[6];};

T Value;

ELEMENT<T>\* Knot;

public:

template<typename ClassName>

bool PrintTheStucture(ClassName\*\* Element) {

if (\*Element != NULL) {

cout << setw(4) << (\*Element)->inf;

for (int i = 0; i < 6; i++)

PrintTheStucture(&((\*Element)->Pointers[i]));

return true;}

return false;}

template<typename Type>

inline void AddElement(Type value) {

NETST::AddElementToStructure(&Knot, value);}};

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

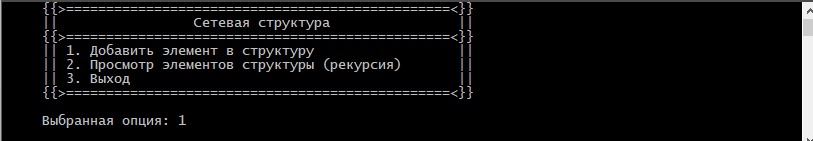
USETSTRUCTURE<int> list;

list.menu();

return 0;}

**Отладка**

Добавление 10 элементов в структуру через меню выбора:





















Рекурсивный вывод введённых элементов структуры на экран:

