Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»

им. В.И. Ульянова (Ленина)»

кафедра Апу

**ОТЧЕТ**

**по лабораторной работе №5**

**«Топологическая сортировка»**

Выполнил: Панфёров М. А.

Группа №3371

Преподаватель: Ряскова Е. Б.

Санкт-Петербург

2024

Постановка задачи

Создать программу для составления учебных планов, в которой:

* реализован диалог с пользователем для ввода числа образовательных дисциплин и связей между ними (пар связанных дисциплин, например, запись [2,1], означает, что для освоения дисциплины 2 требуется освоить дисциплину 1),
* осуществлена проверка возможности корректного размещения всех дисциплин в учебном плане (без нарушения заданной последовательности освоения дисциплин).
* Для взаимодействия с пользователем реализуйте в программе меню, содержащее следующие разделы:
  + Ввод данных о связанных дисциплинах
  + Проверка возможности корректного размещения дисциплин в учебном плане
  + Выход из программы.

Листинг программы

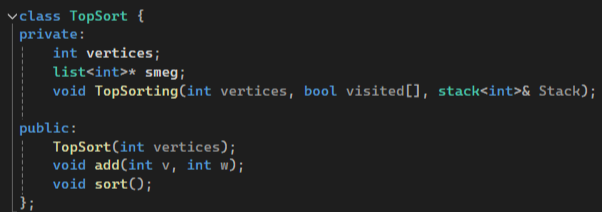


Рис. 1 – Класс топологической сортировки

Класс топологической сортировки содержит в себе данные вершин, список пересечений и функцию сортировки. В общедоступной части доступны функции добавления и вывода.

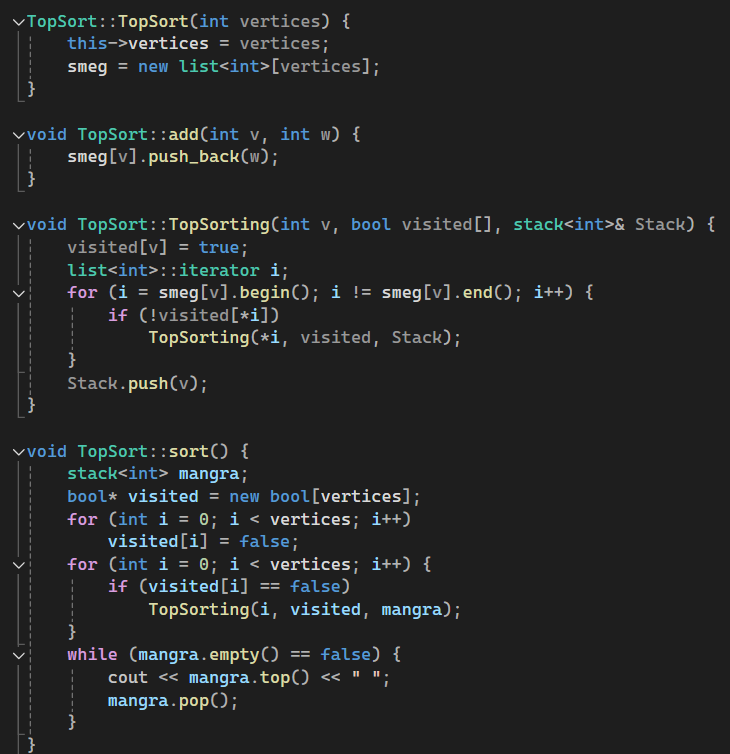


Рис. 2 – Функции топологической сортировки

Выше показаны функции топологической сортировки. В самом верху конструкор, определяющий значения по умолчанию, после идёт функция добавления элемента, после сортировка, а в конце вывод.

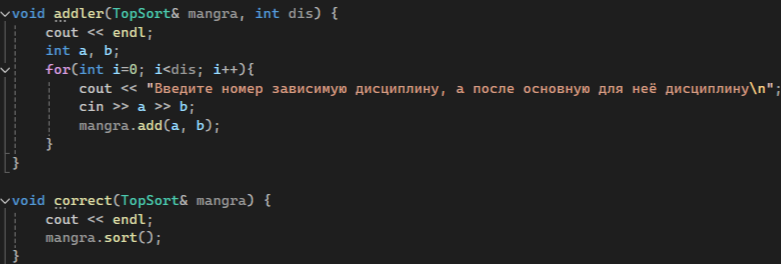


Рис. 3 – Функции ввода и вывода

Функция «addler» позволяет добавлять в наш массив дисциплины (как основные, так и зависимые). В свою очередь «correct» позволяет обратиться к функции вывода.

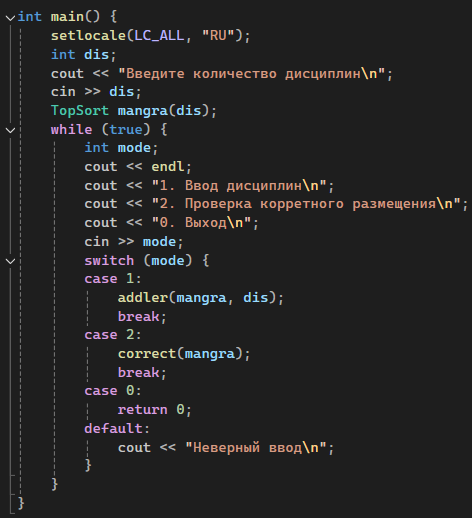


Рис. 4 – Главная функция

В главной функции создаётся массив наших дисциплин (для работы требуется ввести их количество). Далее находится меню, откуда можно перейти и запустить указанные в пункте функции.

Код и пример работы программы

AiSD\_Laba6.cpp:

#include "iostream"

#include "stack"

#include "list"

using namespace std;

class TopSort {

private:

int vertices;

list<int>\* smeg;

void TopSorting(int vertices, bool visited[], stack<int>& Stack);

public:

TopSort(int vertices);

void add(int v, int w);

void sort();

};

TopSort::TopSort(int vertices) {

this->vertices = vertices;

smeg = new list<int>[vertices];

}

void TopSort::add(int v, int w) {

smeg[v].push\_back(w);

}

void TopSort::TopSorting(int v, bool visited[], stack<int>& Stack) {

visited[v] = true;

list<int>::iterator i;

for (i = smeg[v].begin(); i != smeg[v].end(); i++) {

if (!visited[\*i])

TopSorting(\*i, visited, Stack);

}

Stack.push(v);

}

void TopSort::sort() {

stack<int> mangra;

bool\* visited = new bool[vertices];

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

visited[i] = false;

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

if (visited[i] == false)

TopSorting(i, visited, mangra);

}

while (mangra.empty() == false) {

cout << mangra.top() << " ";

mangra.pop();

}

}

void addler(TopSort& mangra, int dis) {

cout << endl;

int a, b;

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

cout << "Введите номер зависимую дисциплину, а после основную для неё дисциплину\n";

cin >> a >> b;

mangra.add(a, b);

}

}

void correct(TopSort& mangra) {

cout << endl;

mangra.sort();

}

int main() {

setlocale(LC\_ALL, "RU");

int dis;

cout << "Введите количество дисциплин\n";

cin >> dis;

TopSort mangra(dis);

while (true) {

int mode;

cout << endl;

cout << "1. Ввод дисциплин\n";

cout << "2. Проверка корретного размещения\n";

cout << "0. Выход\n";

cin >> mode;

switch (mode) {

case 1:

addler(mangra, dis);

break;

case 2:

correct(mangra);

break;

case 0:

return 0;

default:

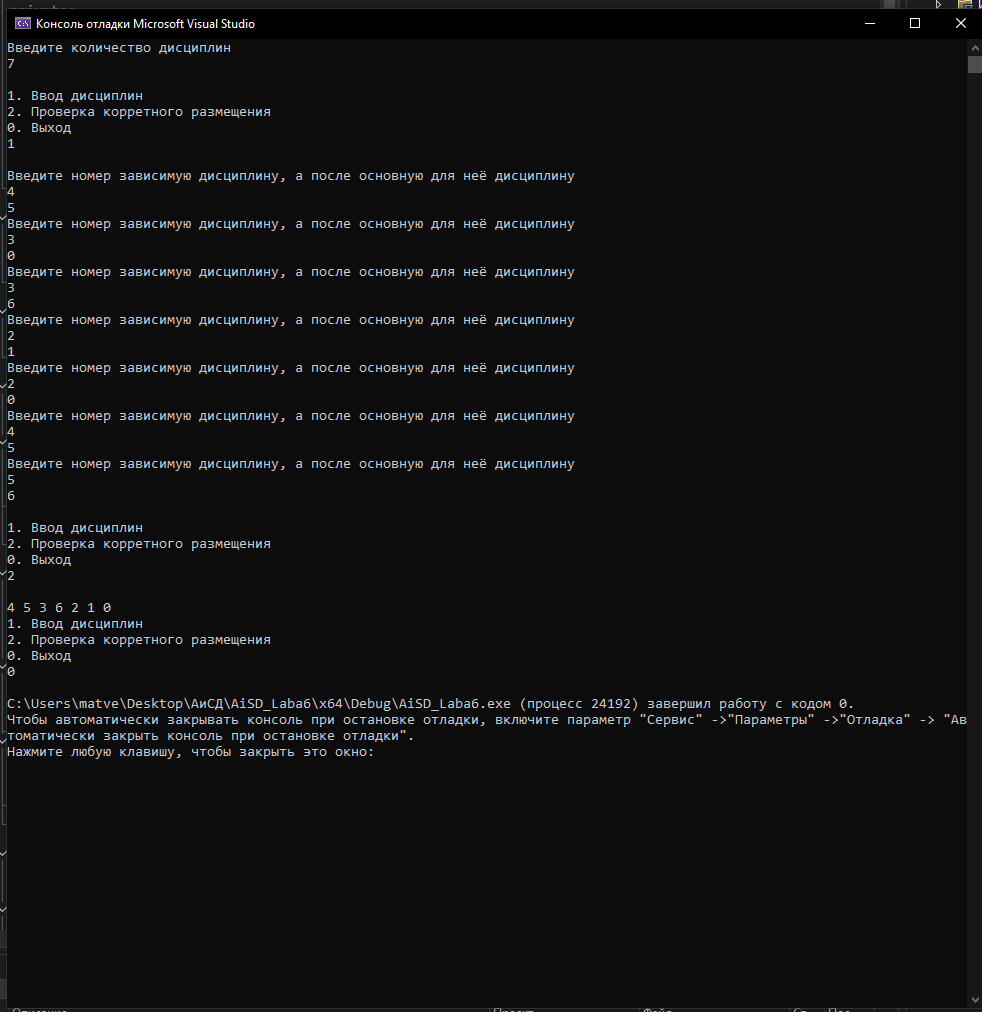
cout << "Неверный ввод\n";

}

}

}

Пример работы программы:



Вывод: я реализовал программу, которая использует топологическую сортировку, и позволяет эффективно строить учебный план. Знаю как она работает и как её использовать. Теперь могу использовать топологическую сортировку в своих проектах.