Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №11.3**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Информационные и динамические структуры. Стэки.

Вариант 23

Выполнил:

Студент группы РИС-20-1б

Кузнецов Михаил Сергеевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

Получить практические навыки работы со стеками.

**Постановка задачи**

1. Сформировать стек. Тип информационного поля double. Добавить в стек после каждого элемента с отрицательным информационным полем элемент с информационным полем равным 0.

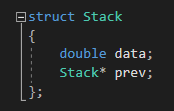
2. Распечатать полученную структуру.

3. Выполнить обработку структуры в соответствии с заданием.

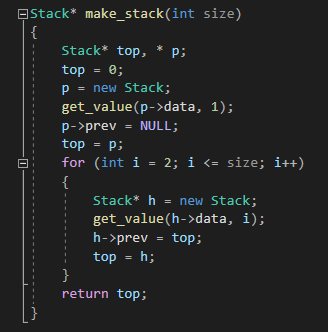
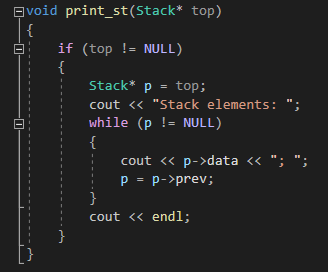
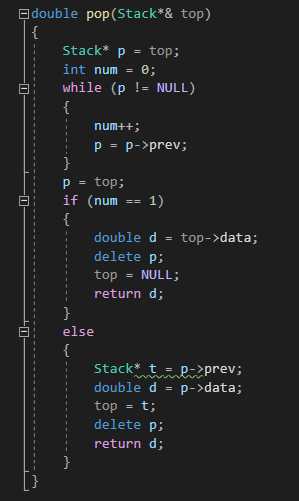
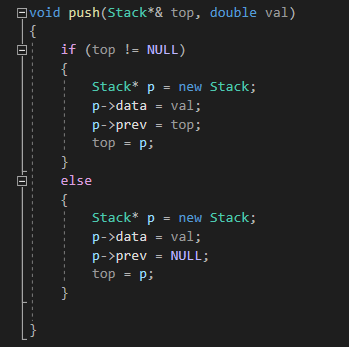
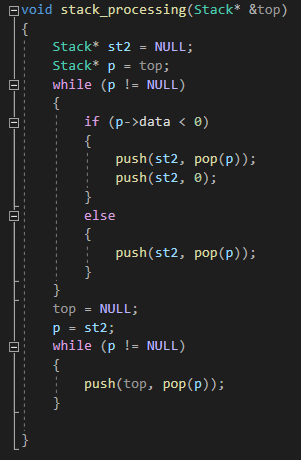
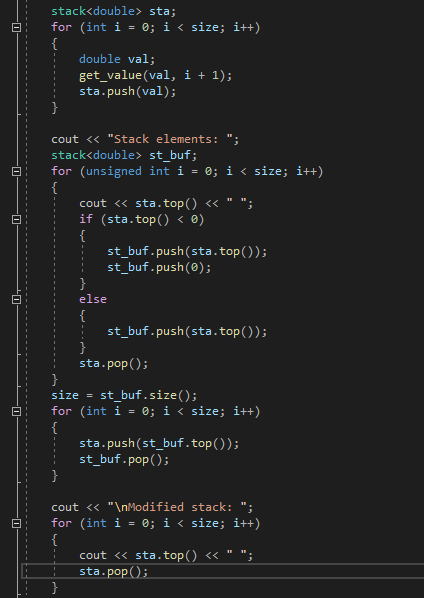
4. Распечатать полученный результат.

5. Удалить соответствующую структуру из памяти.

**Анализ задачи**

1. Определить какие действия необходимо выполнить для решения задачи.
   1. Реализация стэка производится через структуру Stack. 
   2. Создать стэк.



* 1. Заполнить стэк вещественными числами. 
  2. Вывести все элементы стэка в консоль. 
  3. Удаление первого элемента стэка. 
  4. Добавление элемента в стэк. 
  5. Поиск отрицательных значений в стэке и добавление 0. 
  6. Реализация через STL. 

1. Для решения задачи использовались следующие типы данных.

2.1 Данные элементов списка хранятся в формате int.



2.2 Указатель на следующий элемент является экземпляром структуры.



**Код**

#include <iostream>

#include <stack>

using namespace std;

struct Stack

{

double data;

Stack\* prev;

};

template <typename T>

void get\_value(T& value, int i)

{

bool is\_correct = false;

do

{

cout << "Enter " << i << " element\n";

cin >> value;

if (cin.fail())

{

cout << "\nIncorrect value" << endl << endl;

cin.clear();

cin.ignore(32767, '\n');

}

else

{

is\_correct = true;

cin.ignore(32767, '\n');

}

} while (!is\_correct);

}

Stack\* make\_stack(int size)

{

Stack\* top, \* p;

top = 0;

p = new Stack;

get\_value(p->data, 1);

p->prev = NULL;

top = p;

for (int i = 2; i <= size; i++)

{

Stack\* h = new Stack;

get\_value(h->data, i);

h->prev = top;

top = h;

}

return top;

}

void print\_st(Stack\* top)

{

if (top != NULL)

{

Stack\* p = top;

cout << "Stack elements: ";

while (p != NULL)

{

cout << p->data << "; ";

p = p->prev;

}

cout << endl;

}

}

double pop(Stack\*& top)

{

Stack\* p = top;

int num = 0;

while (p != NULL)

{

num++;

p = p->prev;

}

p = top;

if (num == 1)

{

double d = top->data;

delete p;

top = NULL;

return d;

}

else

{

Stack\* t = p->prev;

double d = p->data;

top = t;

delete p;

return d;

}

}

void push(Stack\*& top, double val)

{

if (top != NULL)

{

Stack\* p = new Stack;

p->data = val;

p->prev = top;

top = p;

}

else

{

Stack\* p = new Stack;

p->data = val;

p->prev = NULL;

top = p;

}

}

void stack\_processing(Stack\* &top)

{

Stack\* st2 = NULL;

Stack\* p = top;

while (p != NULL)

{

if (p->data < 0)

{

push(st2, pop(p));

push(st2, 0);

}

else

{

push(st2, pop(p));

}

}

top = NULL;

p = st2;

while (p != NULL)

{

push(top, pop(p));

}

}

int main()

{

int size;

do

{

cout << "Enter size of stack\n";

cin >> size;

if (cin.fail())

{

cin.clear();

cin.ignore(32767, '\n');

size = -1;

}

cin.ignore(32767, '\n');

} while (size <= 0);

cout << "\nManual stack implementation:\n\n";

Stack\* st = make\_stack(size);

print\_st(st);

stack\_processing(st);

print\_st(st);

cout << "\nSTL stack implementation:\n\n";

stack<double> sta;

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

{

double val;

get\_value(val, i + 1);

sta.push(val);

}

cout << "Stack elements: ";

stack<double> st\_buf;

for (unsigned int i = 0; i < size; i++)

{

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

if (sta.top() < 0)

{

st\_buf.push(sta.top());

st\_buf.push(0);

}

else

{

st\_buf.push(sta.top());

}

sta.pop();

}

size = st\_buf.size();

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

{

sta.push(st\_buf.top());

st\_buf.pop();

}

cout << "\nModified stack: ";

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

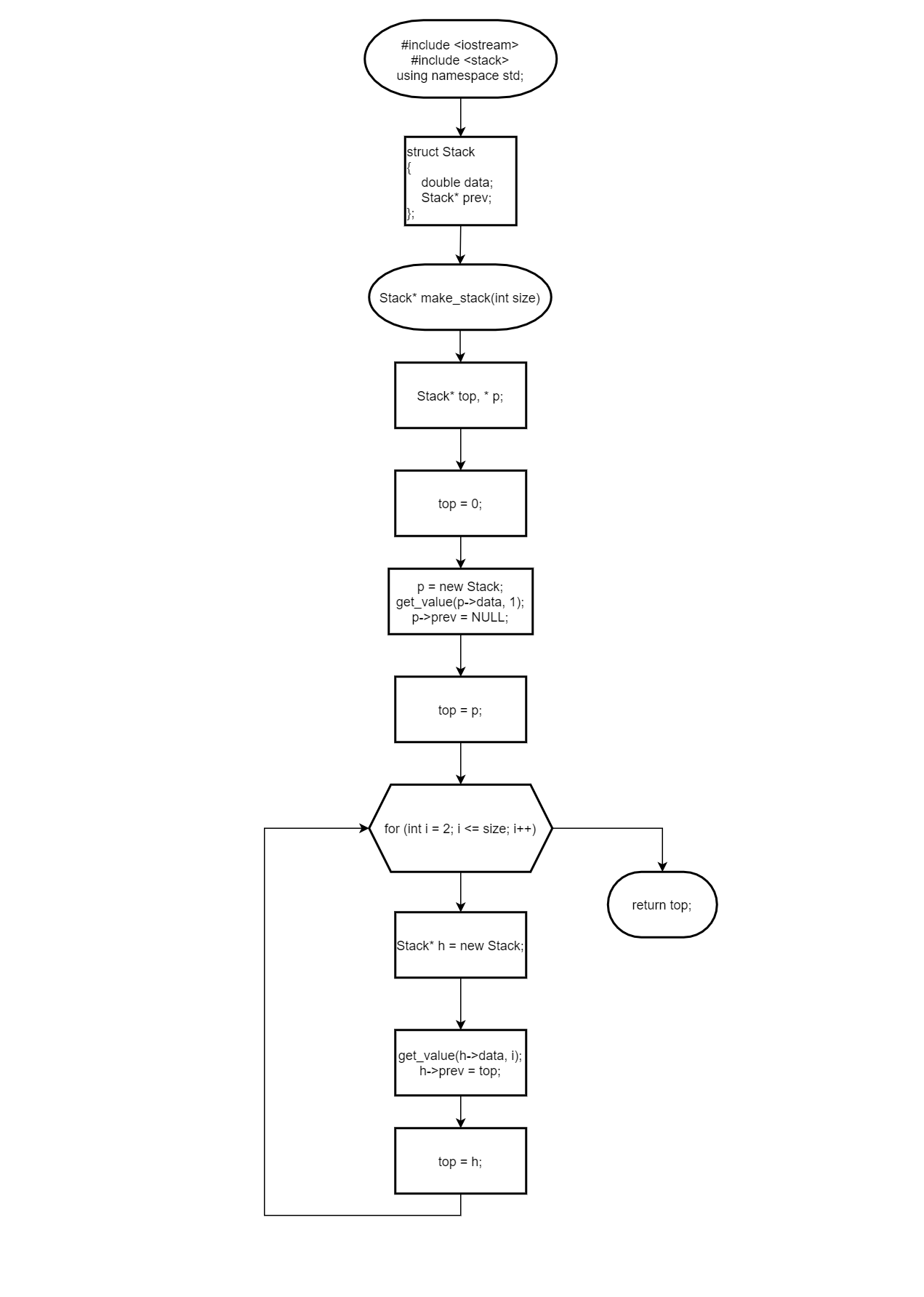
{

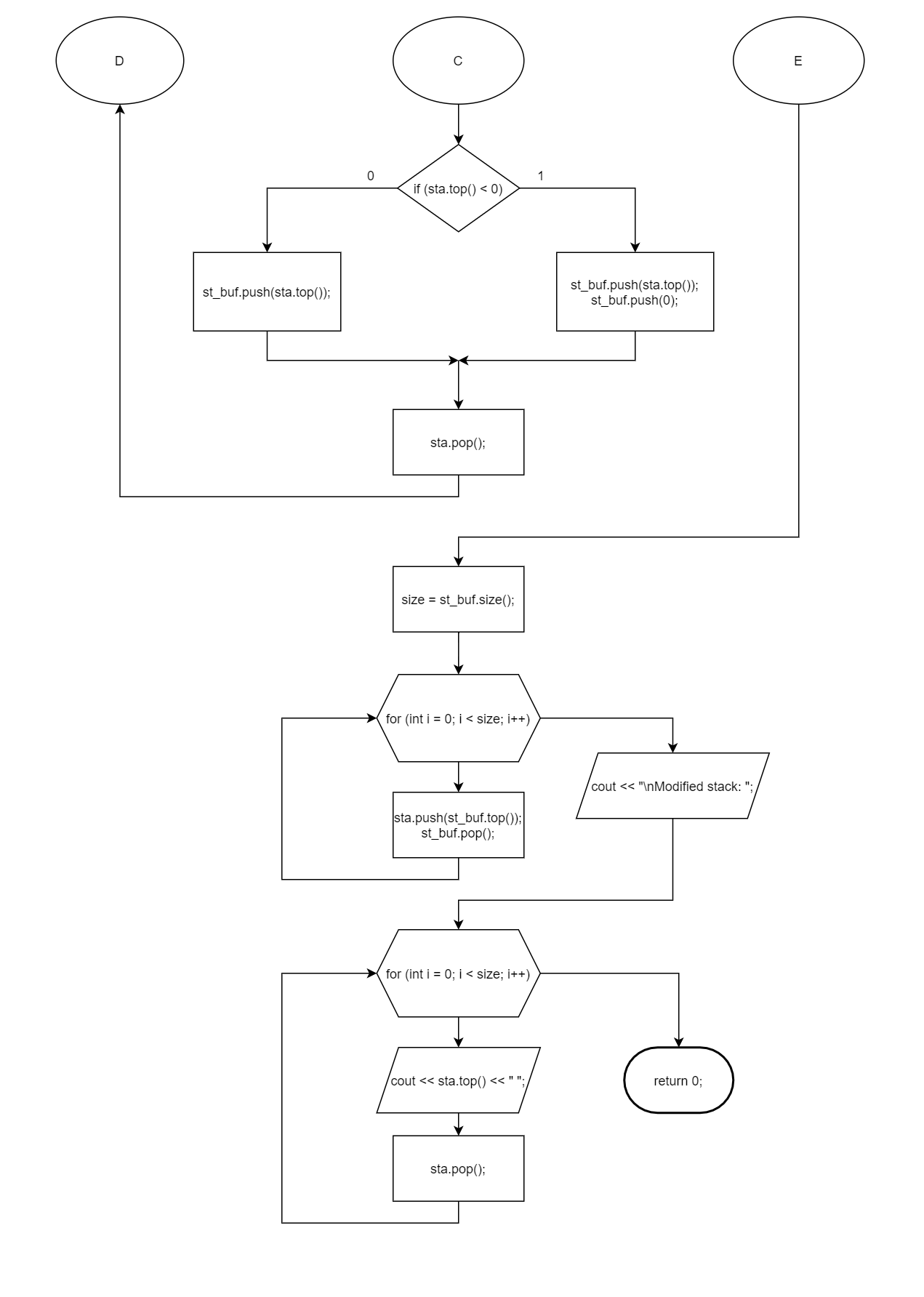
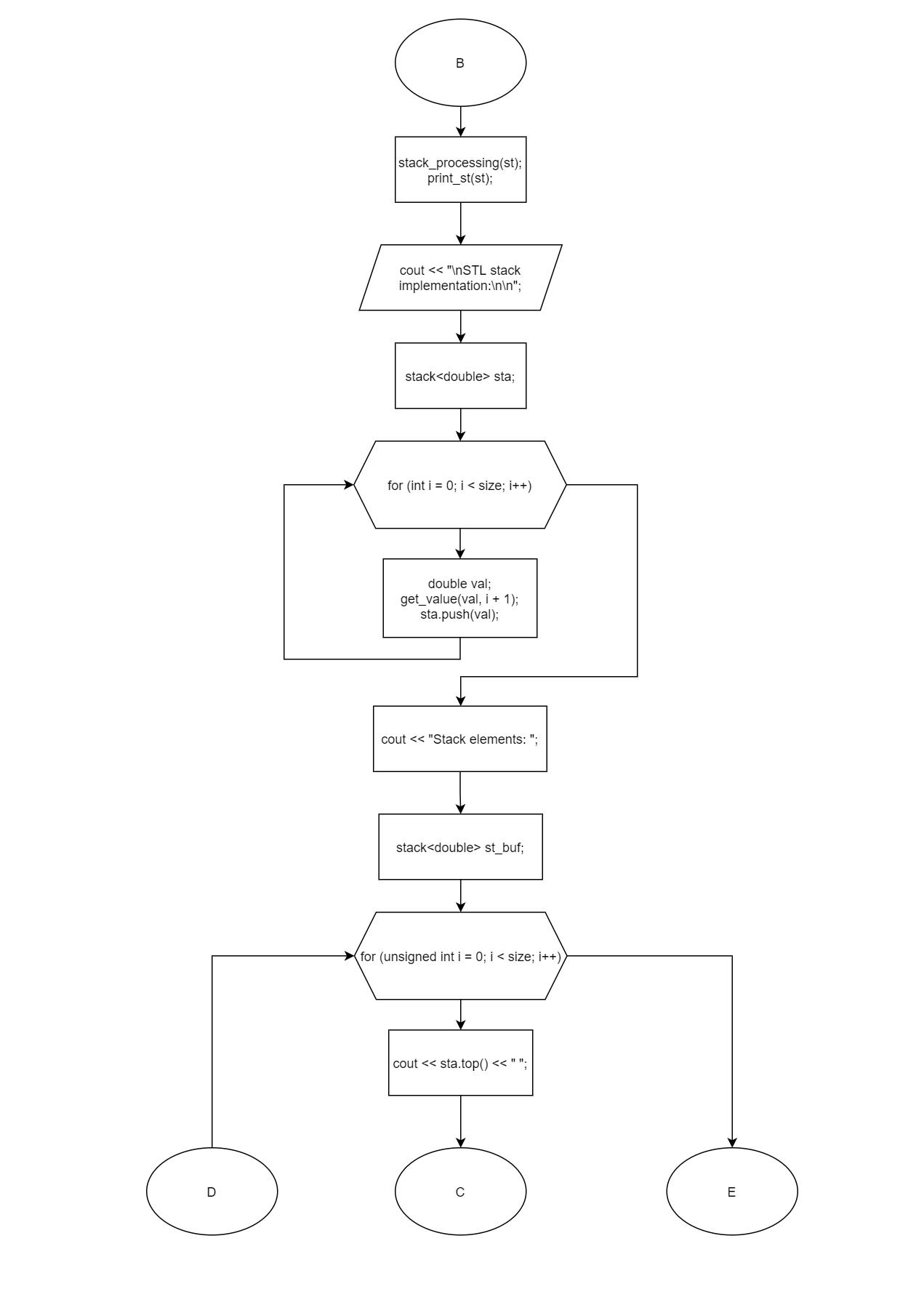
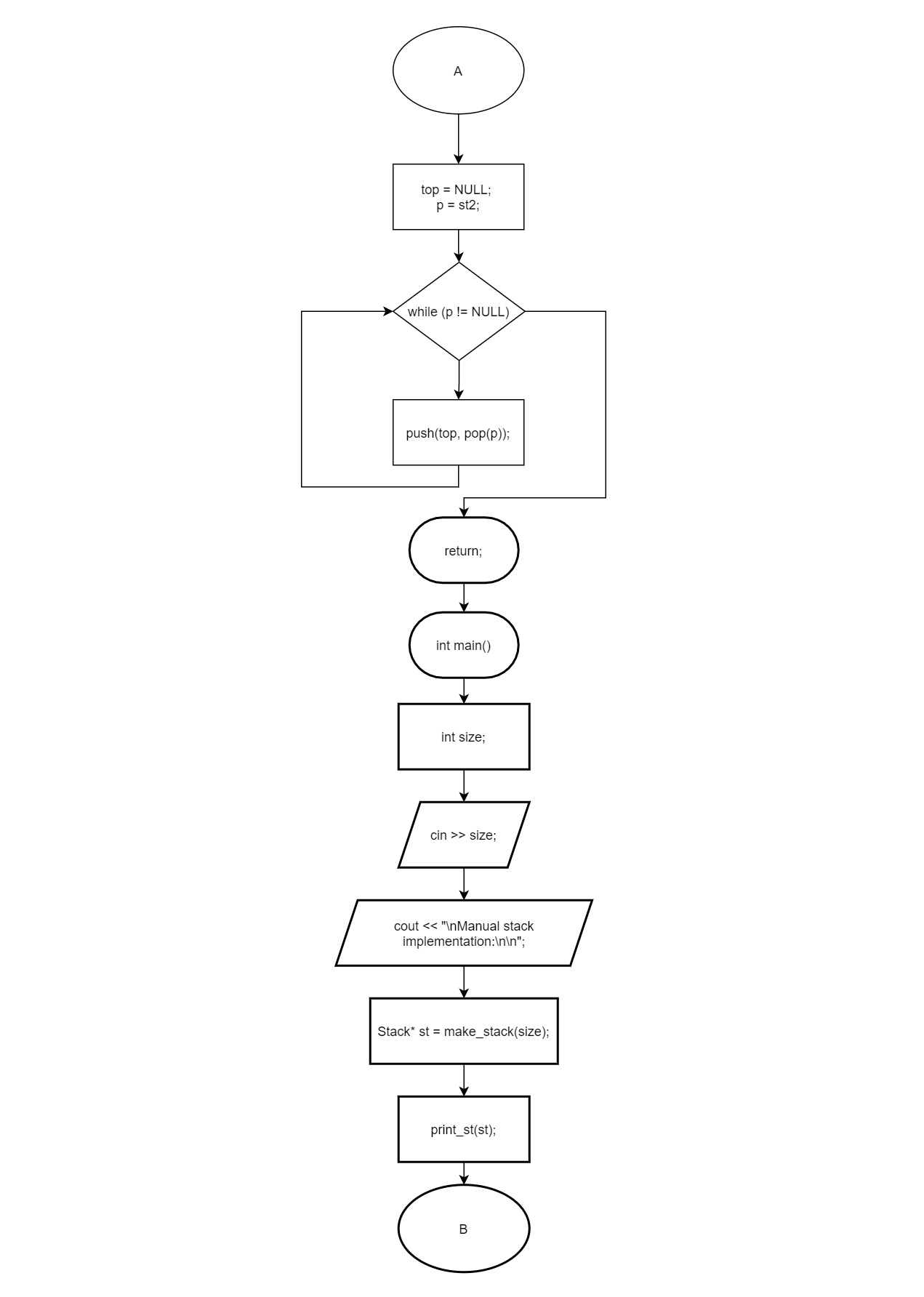
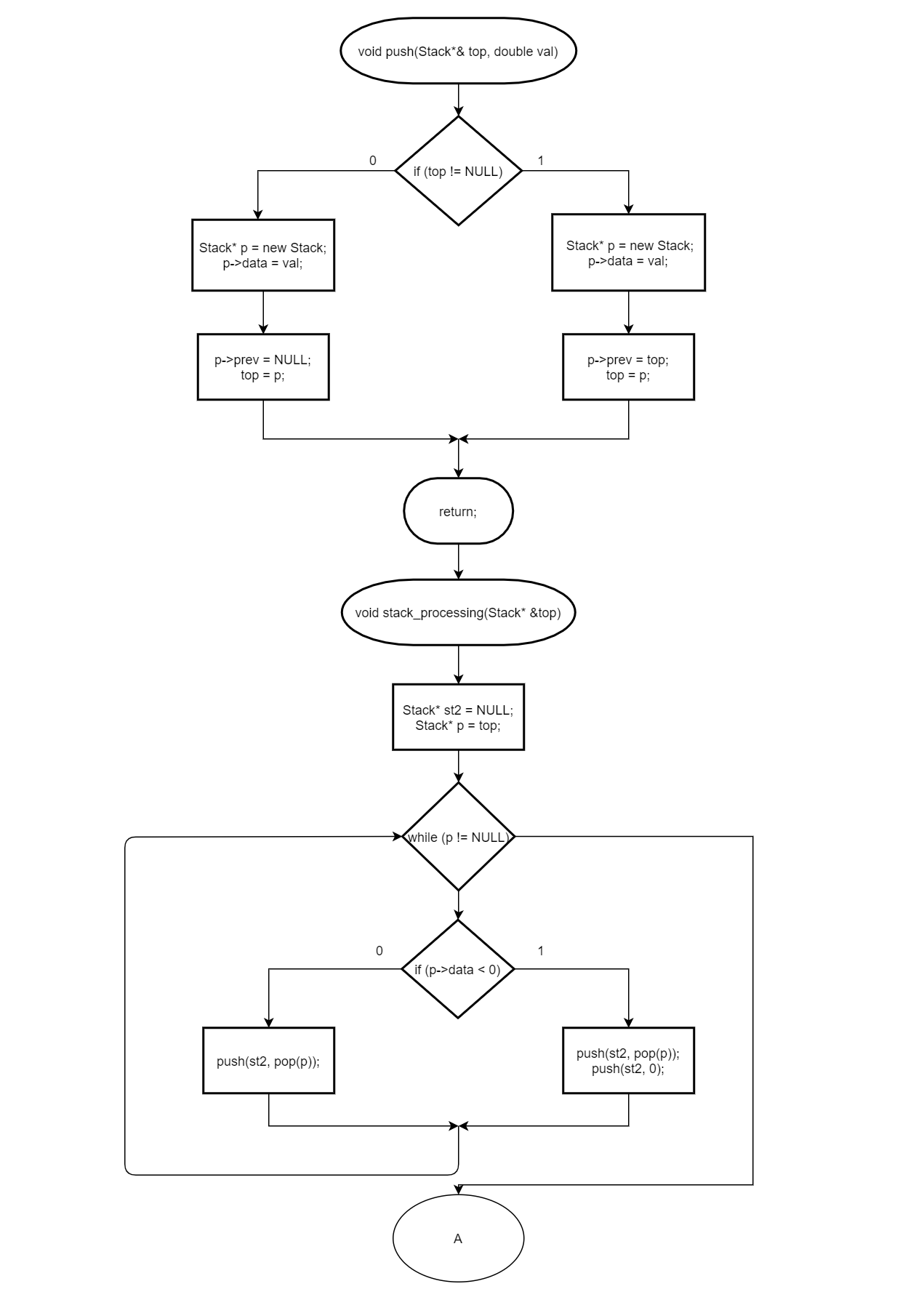
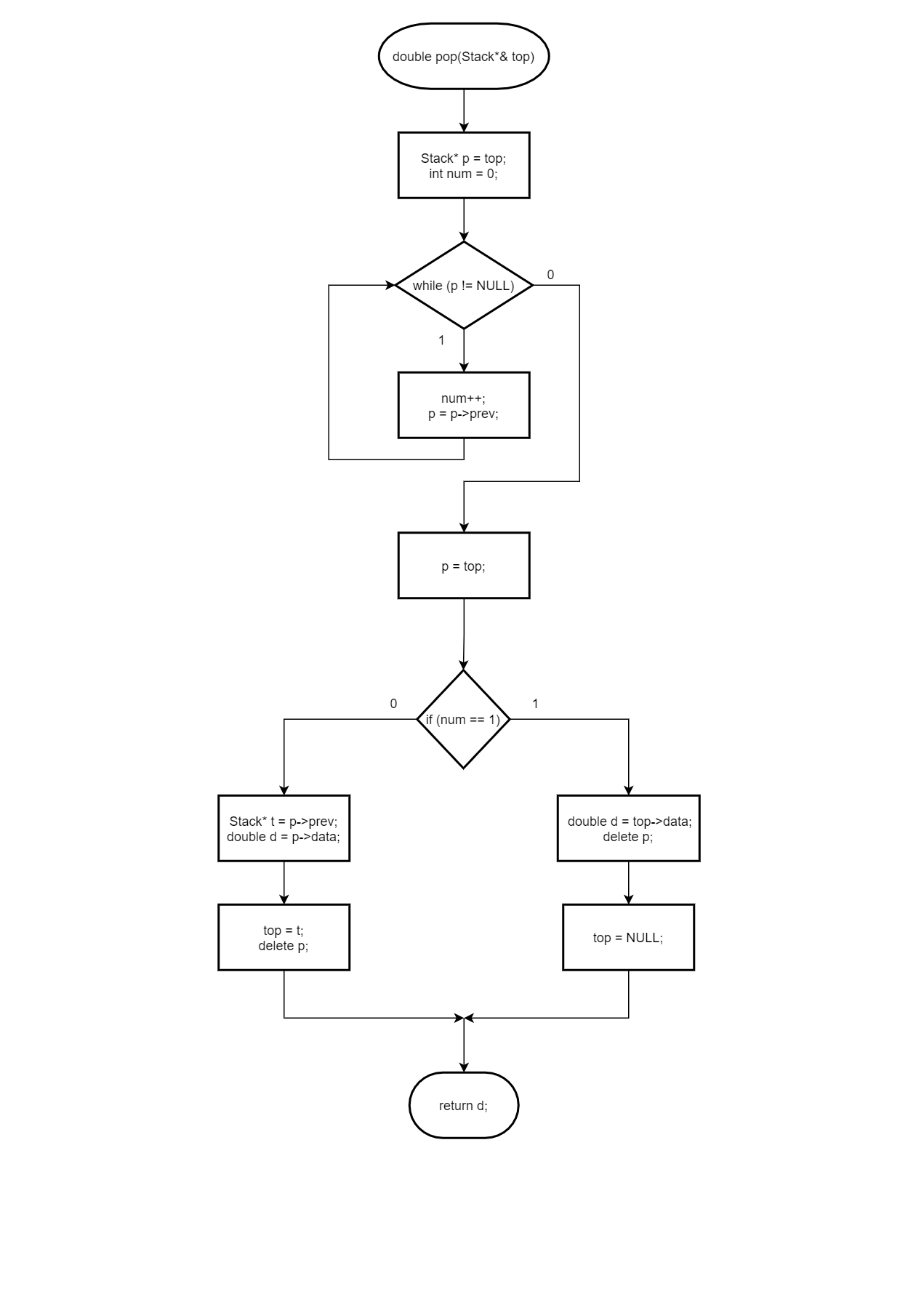
cout << sta.top() << " ";

sta.pop();

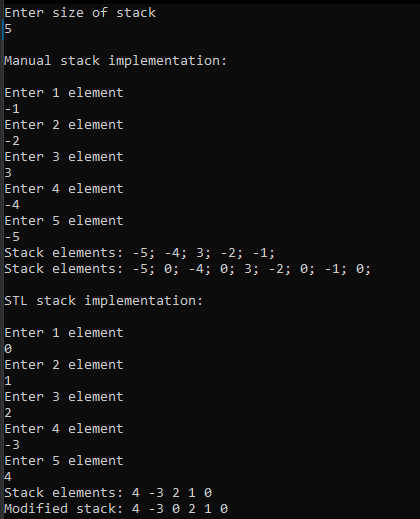
}

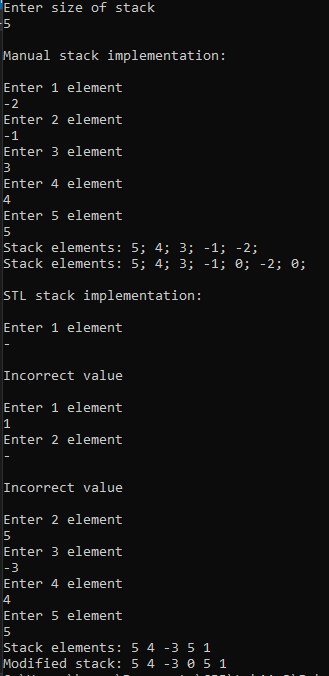
}

**Блок схема **

****

**Тесты**

****

****