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 № 11.2**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Информационные и динамические структуры. Очереди

Вариант 21

Выполнил:

Студент группы РИС-20-1б

Ознобихин Елисей Андреевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

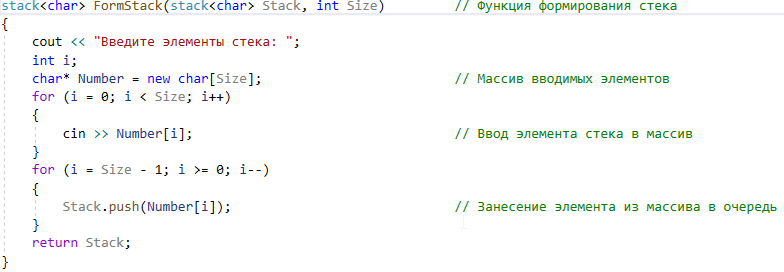
Получить практические навыки работы со стеками.

**Постановка задачи**

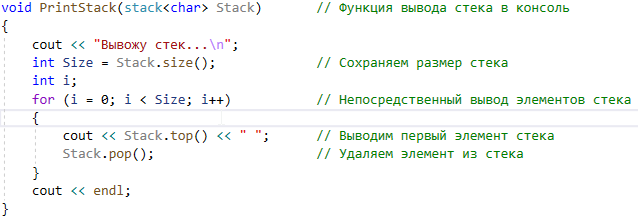
Тип информационного поля char\*. Добавить в стек элемент после элемента с заданным информационным полем.

**STL-библиотека**

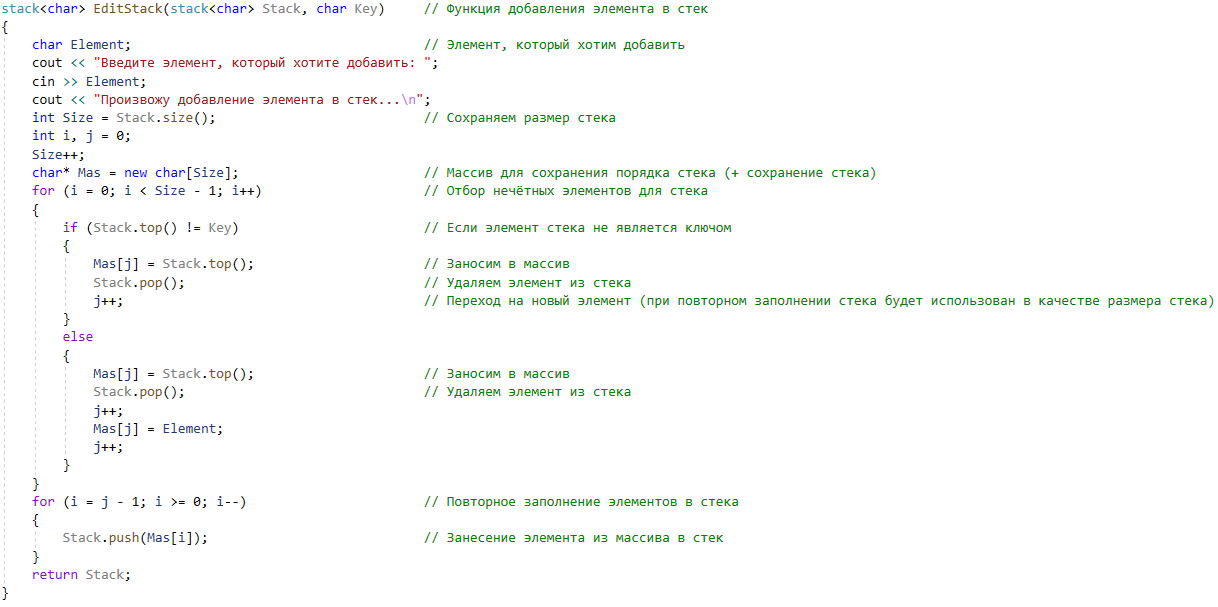
1. Для решения задачи необходимо:
   1. Разработать функцию для формирования стека через stl-библиотеки.



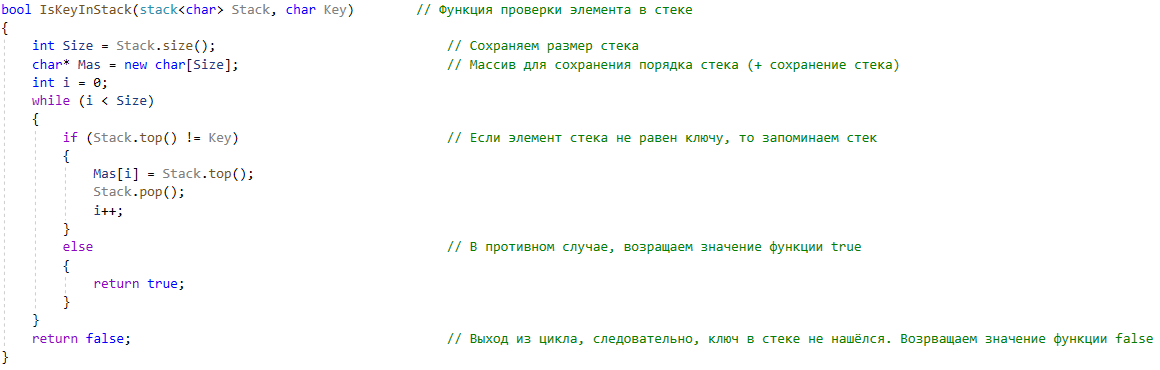
* 1. Разработать функцию для вывода информационного поля стека на экран через консоль.



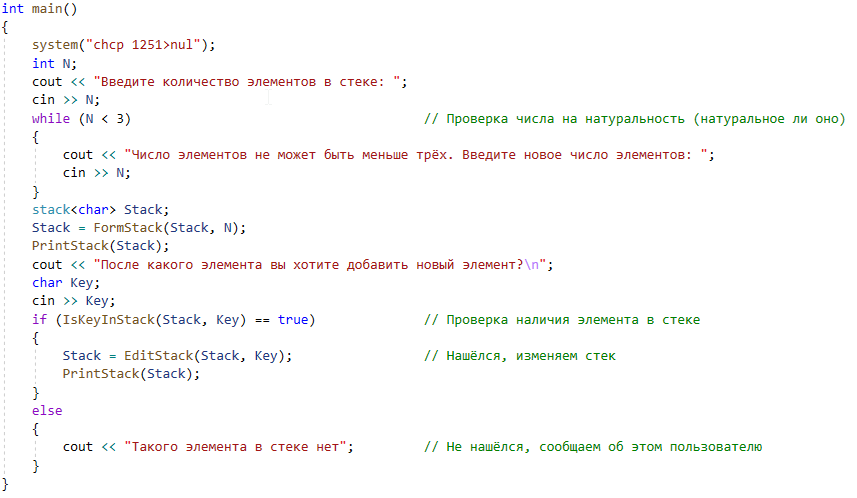
* 1. Разработать функцию для добавления нового элемента после определённого в стек.



* 1. Разработать функцию для нахождения ключа в стеке.



* 1. Реализовать применение этих функций в программе.



1. В ходе работы были использованы типы данных:
   1. Для функции FormStack использовались следующие аргументы:
   2. Тип stack<char>: стек.
   3. Тип int: Размер стека.

Сама функция имеет тип stack<char> и возвращает стек.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_fsQ75lnI2v.png

* 1. Для функции PrintStack использовались следующие аргументы:
  2. Тип stack<char>: стек.

Сама функция имеет тип void, поскольку при работе функции ничего не нужно возвращать.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_OVSWftZdYT.png

* 1. Для функции EditStack использовались следующие аргументы:
  2. Тип stack<char>: стек.
  3. Тип char: ключ поиска.

Сама функция имеет тип stack<char> и возвращает стек.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_uzd6jPS7pw.png

* 1. Для функции IsKeyInStack использовались следующие аргументы:
  2. Тип stack<char>: стек.
  3. Тип char: ключ поиска элемента в стеке.

Сама функция имеет тип bool и возвращает значение истина, если ключ нашелся в списке, или ложь, если ключа в списке нет.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_cgj9z8nbX0.png

1. Для решения задачи данные были представлены в следующем виде:
   1. Для осуществления работы данные были записаны ручным вводом с консоли в виде структур: однонаправленные и двунаправленные списки, стек и очередь.
2. Для операций ввода и вывода данных использовались следующие операторы и функции:
   1. Для вывода информации на консоль использовался оператор cout.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_Fpw1ef0xpt.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_JMK5EqYzIp.png

* 1. Для ввода данных через консоль использовался оператор cin.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_MaXQm3TaYp.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_d5CIhDC2W5.png

1. Поставленные задачи решены следующими действиями:
   1. Все действия со стеком реализованы через написанные вручную функции.

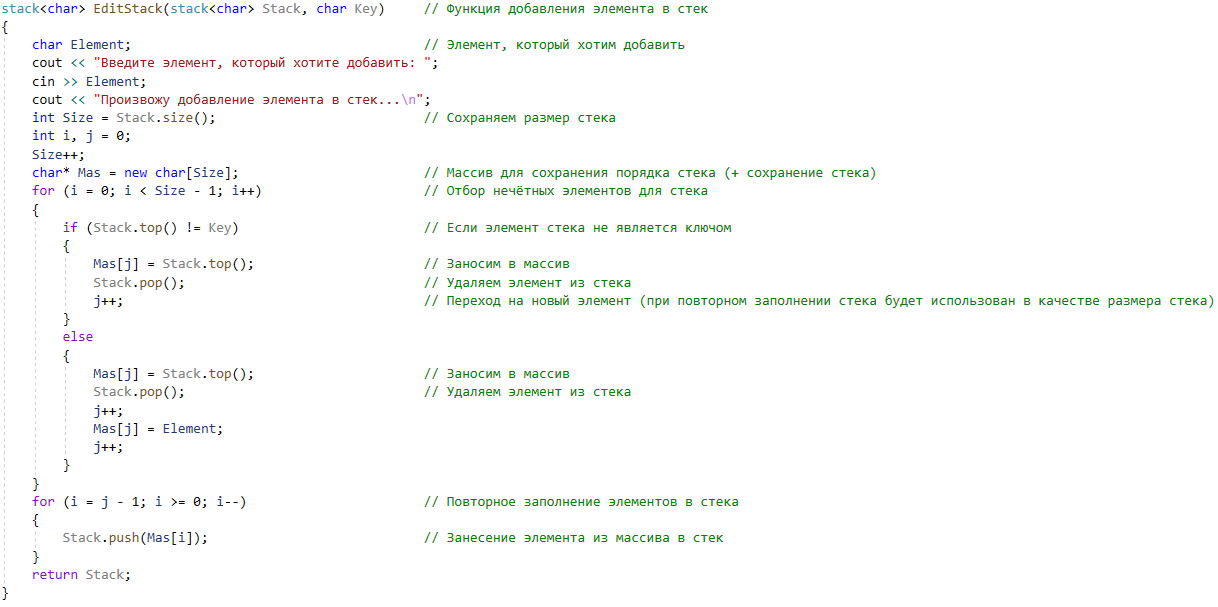
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_fsQ75lnI2v.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_uzd6jPS7pw.png

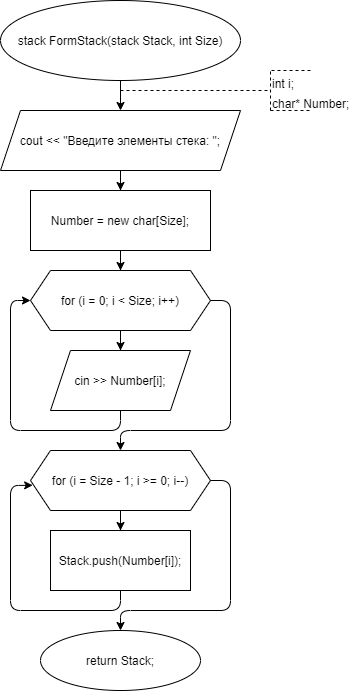
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_cgj9z8nbX0.png

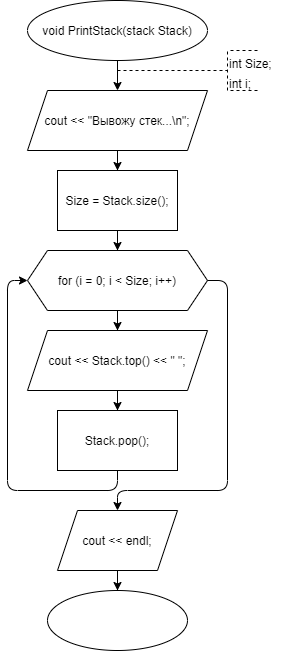
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_OVSWftZdYT.png

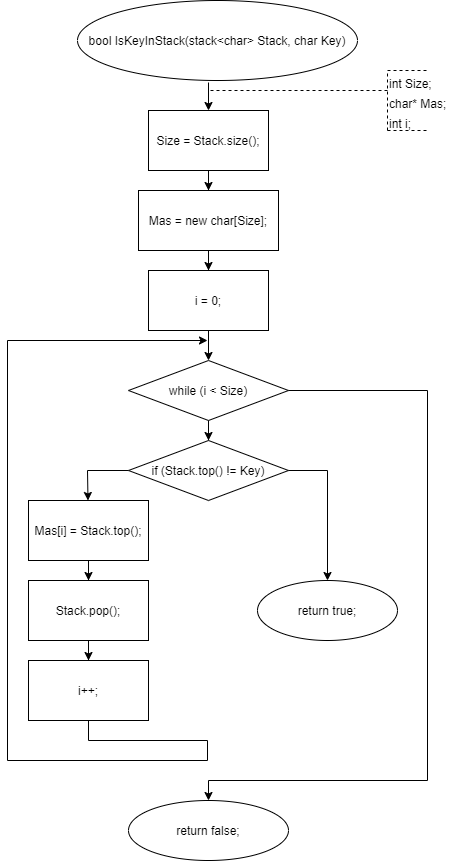
* 1. Добавление элемента после определённого элемента в стеке реализовано с помощью цикла while и вспомогательных функций поиска ключа во множестве.

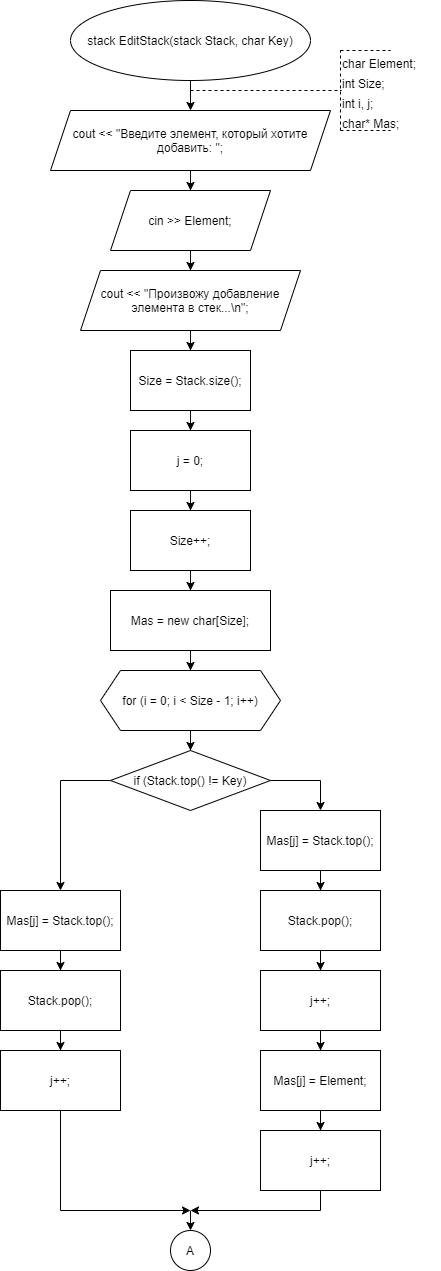


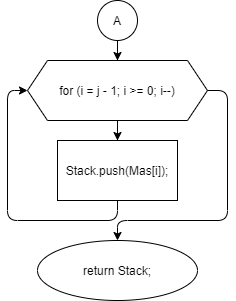
**Блок-схема программы**

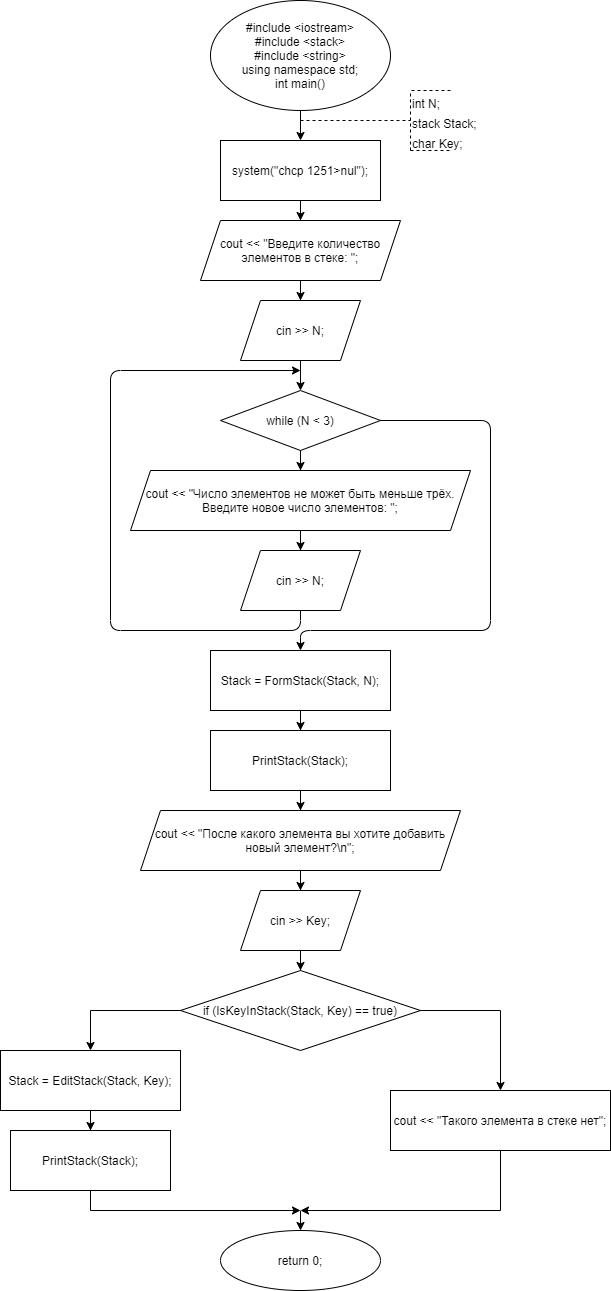
****

****

****

****

****

****

**Реализация задачи на языке С++**

#include <iostream>

#include <stack>

#include <string>

using namespace std;

stack<char> FormStack(stack<char> Stack, int Size)

{

cout << "Введите элементы стека: ";

int i;

char\* Number = new char[Size];

for (i = 0; i < Size; i++)

{

cin >> Number[i];

}

for (i = Size - 1; i >= 0; i--)

{

Stack.push(Number[i]);

return Stack;

}

void PrintStack(stack<char> Stack)

{

cout << "Вывожу стек...\n";

int Size = Stack.size();

int i;

for (i = 0; i < Size; i++)

{

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

Stack.pop();

}

cout << endl;

}

bool IsKeyInStack(stack<char> Stack, char Key)

{

int Size = Stack.size();

char\* Mas = new char[Size];

int i = 0;

while (i < Size)

{

if (Stack.top() != Key)

{

Mas[i] = Stack.top();

Stack.pop();

i++;

}

else

{

return true;

}

}

return false;

}

stack<char> EditStack(stack<char> Stack, char Key)

{

char Element;

cout << "Введите элемент, который хотите добавить: ";

cin >> Element;

cout << "Произвожу добавление элемента в стек...\n";

int Size = Stack.size();

int i, j = 0;

Size++;

char\* Mas = new char[Size];

for (i = 0; i < Size - 1; i++)

{

if (Stack.top() != Key)

{

Mas[j] = Stack.top();

Stack.pop();

j++;

}

else

{

Mas[j] = Stack.top();

Stack.pop();

j++;

Mas[j] = Element;

j++;

}

}

for (i = j - 1; i >= 0; i--)

{

Stack.push(Mas[i]);

}

return Stack;

}

int main()

{

system("chcp 1251>nul");

int N;

cout << "Введите количество элементов в стеке: ";

cin >> N;

while (N < 3)

{

cout << "Число элементов не может быть меньше трёх. Введите новое число элементов: ";

cin >> N;

}

stack<char> Stack;

Stack = FormStack(Stack, N);

PrintStack(Stack);

cout << "После какого элемента вы хотите добавить новый элемент?\n";

char Key;

cin >> Key;

if (IsKeyInStack(Stack, Key) == true)

{

Stack = EditStack(Stack, Key);

PrintStack(Stack);

}

else

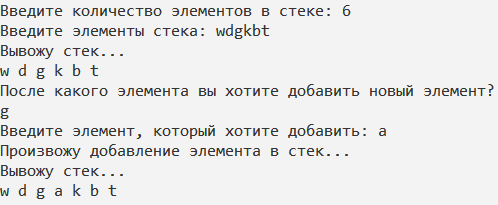
{

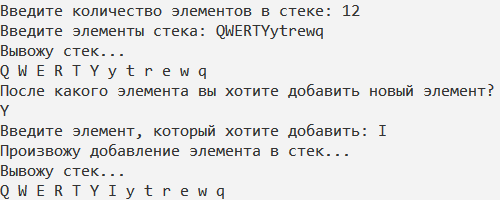
cout << "Такого элемента в стеке нет";

}

}

**Скриншоты результатов работы программы**

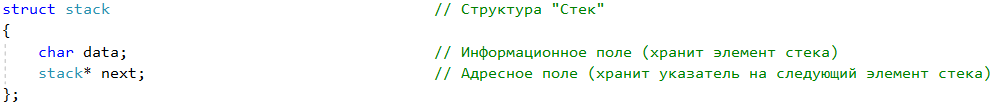
****

****

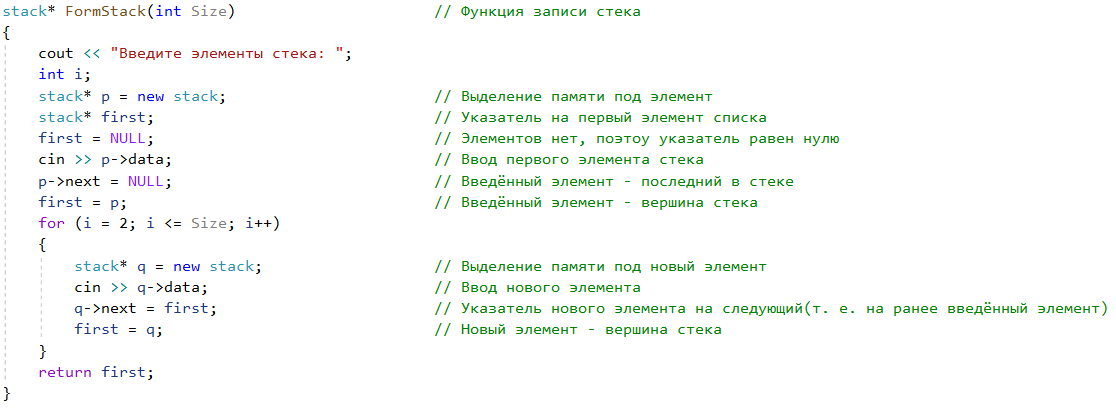
**Cтруктура**

**Анализ задачи**

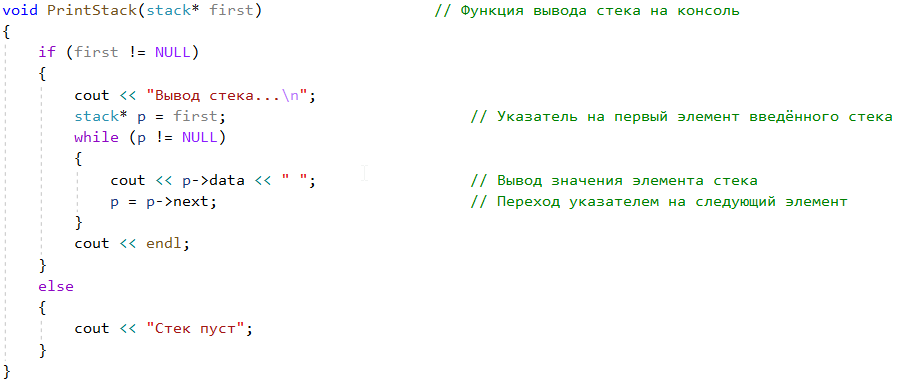
1. Для решения задачи необходимо:
   1. Зарезервировать память под структуру-стек, содержащую информационное поле и адресное поле-указатель на следующий элемент.



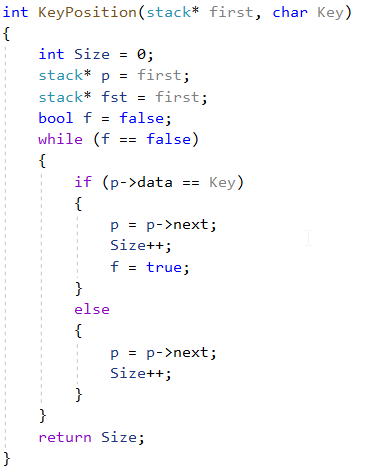
* 1. Разработать функцию для формирования стека через структуру.



* 1. Разработать функцию для вывода информационного поля стека на экран через консоль.



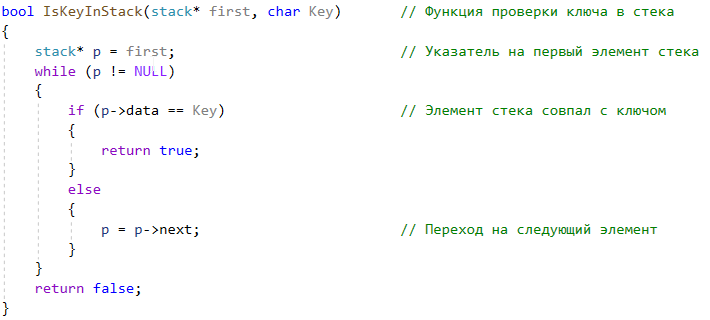
* 1. Разработать функцию для нахождения позиции ключа.



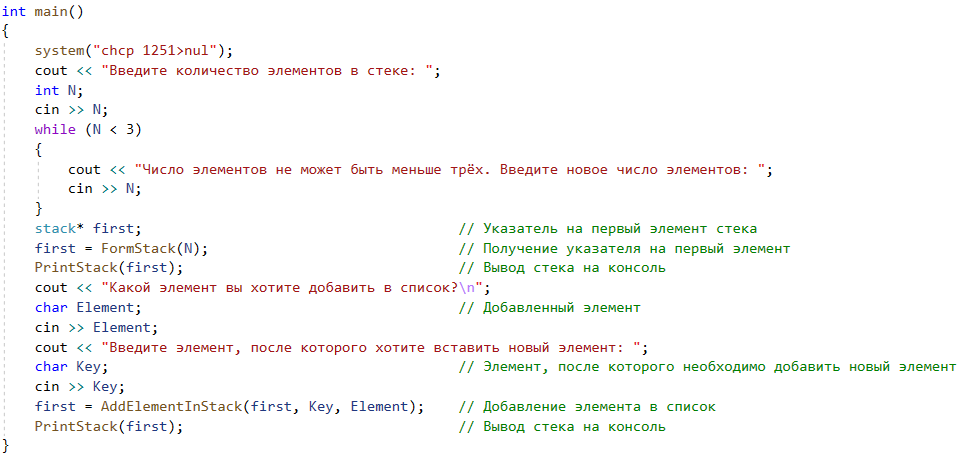
* 1. Разработать функцию для добавления нового элемента после определённого в стек.



* 1. Разработать функцию для нахождения ключа в стеке.



* 1. Реализовать применение этих функций в программе.



1. В ходе работы были использованы типы данных:
   1. Для функции FormStack использовались следующие аргументы:
   2. Тип int: Размер стека.

Сама функция имеет тип указателя на stack и возвращает указатель на вершину стека.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_zWETp9NZet.png

* 1. Для функции PrintStack использовались следующие аргументы:
  2. Указатель на stack: Указатель на вершину стека.

Сама функция имеет тип void, поскольку при работе функции ничего не нужно возвращать.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_obTCYDpXYV.png

* 1. Для функции KeyPosition использовались следующие аргументы:
  2. Указатель на queue: Указатель на начало очереди.
  3. Тип char: ключ поиска.

Сама функция имеет тип int и возвращает позицию найденного ключа.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_8YRUAwfBDp.png

* 1. Для функции AddElementInStack использовались следующие аргументы:
  2. Указатель на stack: указатель на вершину стека.
  3. Тип char: ключ поиска элемента в стеке.
  4. Тип char: добавленный элемент в стек.

Сама функция имеет тип указателя на stack и возвращает указатель на вершину стека.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_LCVMoHNaYu.png

* 1. Для функции IsKeyInStack использовались следующие аргументы:
  2. Указатель на list: указатель на первый элемент списка.
  3. Тип char: ключ поиска элемента в списке.

Сама функция имеет тип bool и возвращает значение истина, если ключ нашелся в списке, или ложь, если ключа в списке нет.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\ApplicationFrameHost_F1EAMd7VkM.png

1. Для решения задачи данные были представлены в следующем виде:
   1. Для осуществления работы данные были записаны ручным вводом с консоли в виде структуры-стека, которая хранит в себе информационное поле и адресное поле.
2. Для операций ввода и вывода данных использовались следующие операторы и функции:
   1. Для вывода информации на консоль использовался оператор cout.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_aEPXSXqDf8.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_NZuMkD0GWK.png

* 1. Для ввода данных через консоль использовался оператор cin.



1. Поставленные задачи решены следующими действиями:
   1. Все действия со стеком реализованы через написанные вручную функции.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_LCVMoHNaYu.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\ApplicationFrameHost_F1EAMd7VkM.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_8YRUAwfBDp.png

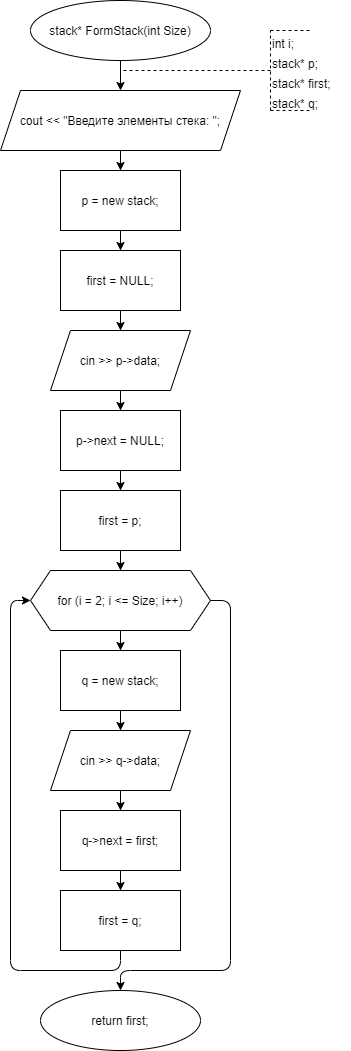
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_obTCYDpXYV.png

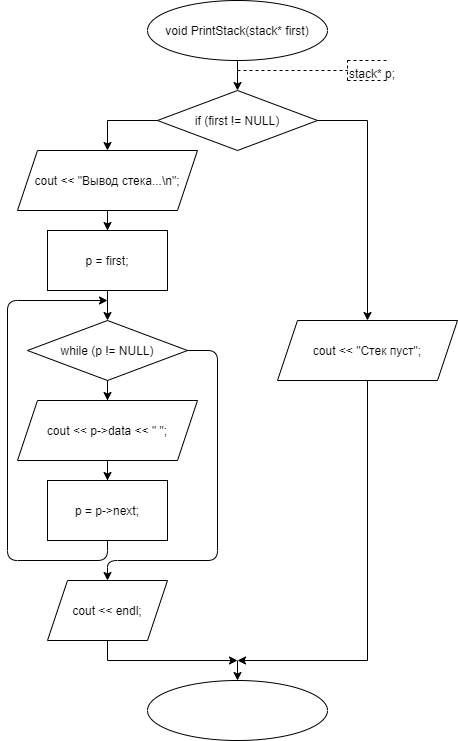
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_zWETp9NZet.png

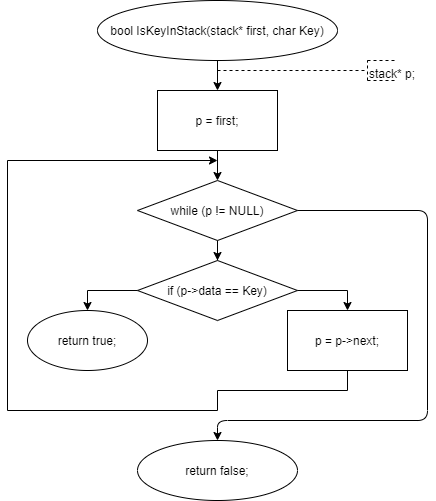
* 1. Добавление элемента после определённого элемента в стеке реализовано с помощью цикла while и вспомогательных функций поиска ключа во множестве.

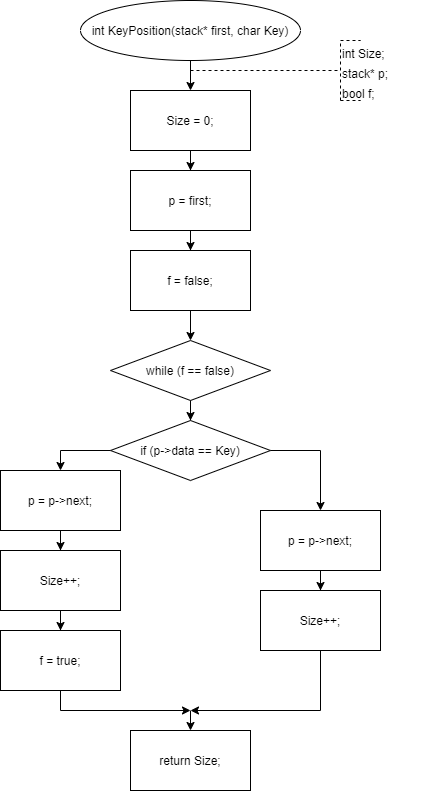


**Блок-схема программы**

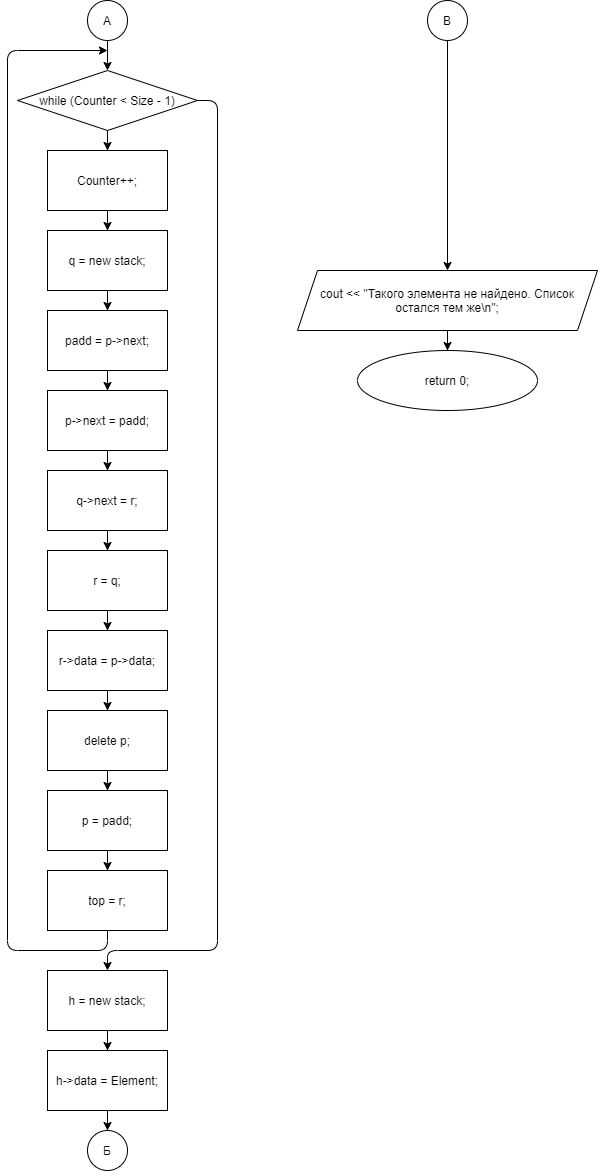


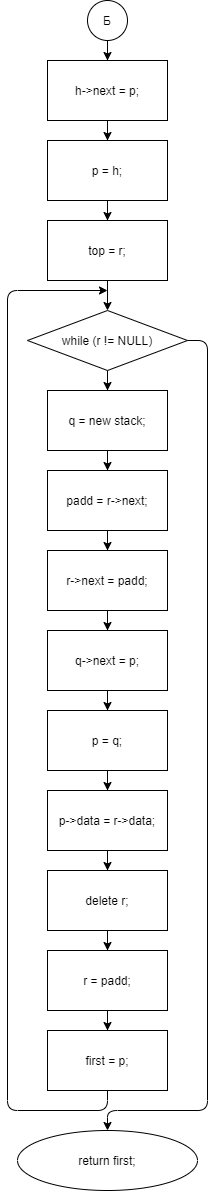


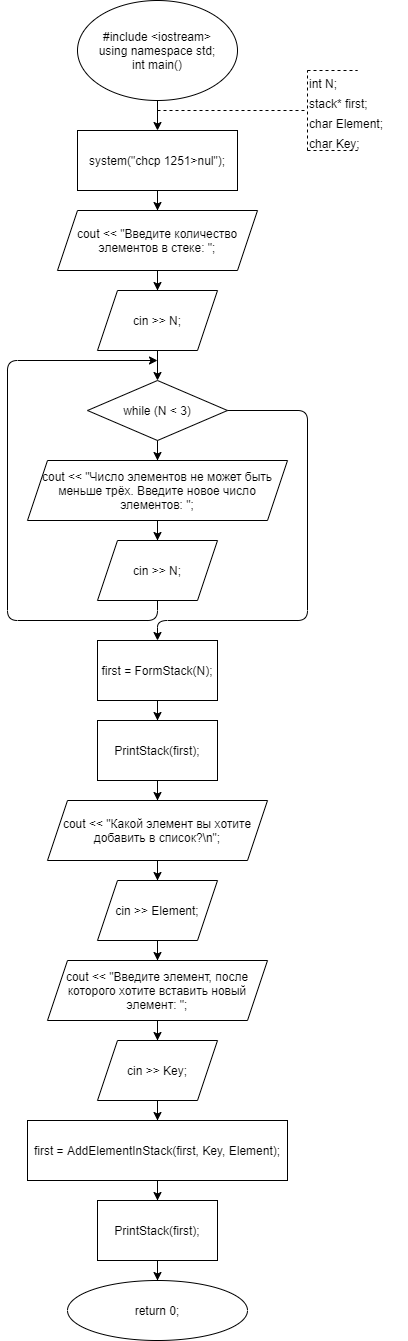












**Реализация задачи на языке С++**

#include <iostream>

using namespace std;

struct stack

{

char data;

stack\* next;

};

stack\* FormStack(int Size)

{

cout << "Введите элементы стека: ";

int i;

stack\* p = new stack;

stack\* first;

first = NULL;

cin >> p->data;

p->next = NULL;

first = p;

for (i = 2; i <= Size; i++)

{

stack\* q = new stack;

cin >> q->data;

q->next = first;

first = q;

}

return first;

}

void PrintStack(stack\* first)

{

if (first != NULL)

{

cout << "Вывод стека...\n";

stack\* p = first;

while (p != NULL)

{

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

p = p->next;

}

cout << endl;

}

else

{

cout << "Стек пуст";

}

}

bool IsKeyInStack(stack\* first, char Key)

{

stack\* p = first;

while (p != NULL)

{

if (p->data == Key)

{

return true;

}

else

{

p = p->next;

}

}

return false;

}

int KeyPosition(stack\* first, char Key)

{

int Size = 0;

stack\* p = first;

bool f = false;

while (f == false)

{

if (p->data == Key)

{

p = p->next;

Size++;

f = true;

}

else

{

p = p->next;

Size++;

}

}

return Size;

}

stack\* AddElementInStack(stack\* first, char Key, char Element)

{

if (IsKeyInStack(first, Key) == true)

{

stack\* p = first;

stack\* r = new stack;

stack\* top = NULL;

int Counter = 0, Size;

Size = KeyPosition(first, Key);

stack\* p1 = p->next;

p->next = p1;

r->data = p->data;

delete p;

p = p1;

r->next = NULL;

top = r;

while (Counter < Size - 1)

{

Counter++;

stack\* q = new stack;

stack\* padd = p->next;

p->next = padd;

q->next = r;

r = q;

r->data = p->data;

delete p;

p = padd;

top = r;

}

stack\* h = new stack;

h->data = Element;

h->next = p;

p = h;

top = r;

while (r != NULL)

{

stack\* q = new stack;

stack\* padd = r->next;

r->next = padd;

q->next = p;

p = q;

p->data = r->data;

delete r;

r = padd;

first = p;

}

return first;

}

else

{

cout << "Такого элемента не найдено. Список остался тем же\n";

return 0;

}

}

int main()

{

system("chcp 1251>nul");

cout << "Введите количество элементов в стеке: ";

int N;

cin >> N;

while (N < 3)

{

cout << "Число элементов не может быть меньше трёх. Введите новое число элементов: ";

cin >> N;

}

stack\* first;

first = FormStack(N);

PrintStack(first);

cout << "Какой элемент вы хотите добавить в список?\n";

char Element;

cin >> Element;

cout << "Введите элемент, после которого хотите вставить новый элемент: ";

char Key;

cin >> Key;

first = AddElementInStack(first, Key, Element);

PrintStack(first);

}

**Скриншоты результатов работы программы**

