**Класс stack**

Объект выделяет и освобождает хранилище для последовательности его элементы управления через основной контейнер, типа *Container*, который хранит элементы *Value* и увеличивается по требованию. Объект ограничивает доступ к элементам по принципу LIFO, последним пришел — первым в доступе.

|  |  |
| --- | --- |
| **Функция Member** | **Описание** |
| [stack::assign (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb385087.aspx) | Заменяет все элементы. |
| [stack::empty (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb385212.aspx) | Тесты отсутствуют ли какие-либо элементы. |
| [stack::get\_container (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb386077.aspx) | Осуществляет доступ к базовому контейнеру. |
| [stack::pop (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb386057.aspx) | Удаляет последний элемент. |
| [stack::push (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb157596.aspx) | Добавляет новый последний элемент. |
| [stack::size (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb157600.aspx) | Подсчитывает количество элементов. |
| [stack::stack (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb386202.aspx) | Создает объект контейнера. |
| [stack::top (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb385571.aspx) | Получает последний элемент. |
| [stack::to\_array (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb386120.aspx) | Копирует управляемую последовательность в новый массив. |

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| [stack::top\_item (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb384986.aspx) | Получает последний элемент. |
| **operator** | **Описание** |
| [stack::operator= (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb157682.aspx) | Заменяет контролируемая последовательность. |
| [operator!= (stack) (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb398133.aspx) | Определяет, является ли объект **stack** не равен другому объекту **stack**. |
| [operator< (stack) (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb157754.aspx) | Определяет, является ли объект **stack** меньше другой объект **stack**. |
| [operator<= (stack) (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb157855.aspx) | Определяет, является ли объект **stack** меньше или равно другому объекту **stack**. |
| [operator== (stack) (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb385651.aspx) | Определяет, является ли объект **stack** равен другому объекту **stack**. |
| [operator> (stack) (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb157748.aspx) | Определяет, является ли объект **stack** больше другого объекта **stack**. |
| [operator>= (stack) (STL/CLR)](https://msdn.microsoft.com/ru-ru/library/bb398129.aspx) | Определяет, является ли объект **stack** больше или равно другому объекту **stack**. |

//Игнащенко Татьяна 8 группа

//1.2 Подсчет количество элементов стека, у которых равные "соседи".

//Тесты: "Input.txt" 2 1 2 3 3 4 3 4 5 4 6 7 "Input. txt" 1 1 1 2 3 2 4 4 5 8 8 5 5 5 6 5

// Результат: 5 3 4 1 Результат: 6 5 3 1

#include <iostream>

#include <fstream>

#include <stack>

using namespace std;

void main()

{

setlocale(LC\_ALL, ".1251");

fstream in;

in.open("Input.txt", ios::in);

stack<int> st;

int n;

while (in.good())

{

in >> n; //читаем из файла

st.push(n);//заносим в стек(элемент становится вершиной)

}

int a, b, c;

a = st.top();

st.pop();

b =st.top();

st.pop();

cout << "Результат: ";

int k = 0;

while (!st.empty())

{

int c = st.top();

if (a == c)

{

cout << b << " ";

k++;

}

a = b;

b = c;

st.pop();

}

if (!k) cout << "Нет элементов с одинаковыми соседями";

cout << endl;

system("pause");

}