**Лабораторная работа № 1**

**Шаблоны (параметризованные типы)**

**Выполнил студент группы ИВТ-12М**

**Ковальчук Антон**

**Вариант 9**

**Задание 1.**

****

#include <stdafx.h>

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

int ans[20];

int h = 0;

int arr[20];

int key;

srand(time(NULL));

for (int i = 0; i < 20; i++)

{

arr[i] = 1 + rand() % 20;

cout << arr[i] << " ";

if (i==9)

{

cout << endl;

}

}

cout << endl << endl << "Введите элемент:" ;

cin >> key;

for (int i = 0; i < 20; i++)

{

if (arr[i] == key)

{

ans[h++] = i;

}

}

if (h != 0)

{

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

{

cout << "Элемент " << key << " находится в ячейке " << ans[i] << endl;

}

}

else

{

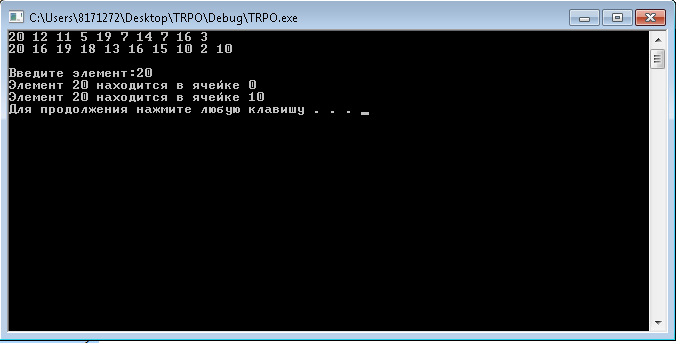
cout << "Мы не нашли элемент" << key << "в массиве" << endl;

}

system("pause");

return 0;

}



На изображении сверху мы видим, что при вводе конкретного элемента массива находится номер этого элемента, при этом, если один и тот же элемент будет встречаться нам несколько раз, программа укажет нам несколько индексов, в которых находится искомый элемент.

**Задание 2.**

****

#include <iostream>

#include <cfenv>

#include <cmath>

int main()

{

std::feclearexcept( FE\_ALL\_EXCEPT );

float z = 123456789012345678901234567890.f \* 12345678900;

if ( std::fetestexcept( FE\_OVERFLOW ) ) {

std::cout << "overflow\n";

} else {

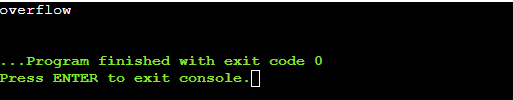
std::cout << "ok\n";

}

}

Данная программа действительно обрабатывает ошибку переполнения по типу и выдаёт в этом случае на экран сообщение overflow

В результате выполнения написанного кода мы увидим следующее:



Два числа настолько велики, что у нас возникает переполнение по типу, о чём и говорится в сообщении, выведенном на экран. Убрав любую одну цифру в первом множителе, получим обратное сообщение о том, что ошибка переполнения по типу отсутствует и у нас всё хорошо.

#include <iostream>

#include <cfenv>

#include <cmath>

int main()

{

std::feclearexcept( FE\_ALL\_EXCEPT );

float z = 12345678901234567890123456789.f \* 12345678900;

if ( std::fetestexcept( FE\_OVERFLOW ) ) {

std::cout << "overflow\n";

} else {

std::cout << "ok\n";

}

}

