САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Практическая работа №14

«Использование шаблонных функций и классов»

Выполнил: Мороз И.О.

Группа К3120

Проверил: Осипов Н.А.

Санкт-Петербург

2018 г.

# Упражнение 1.

1. Была создана функция для сортировки целочисленного массива

void sorting(int arr[], int size) {

int j = 0;

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

{

int x = arr[i];

for (j = i - 1; j >= 0 && x < arr[j]; j--)

{

arr[j + 1] = arr[j];

}

arr[j + 1] = x;

}

}

1. В методе main был создан массив целых чисел, который после был отсортирован функцией sorting()

int arr[] = { 9,3,17,6,5,4,31,2,12 };

int k1 = sizeof(arr) / sizeof(arr[0]);

sorting(arr, k1);

for (int i = 0; i < k1; i++) {

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

}

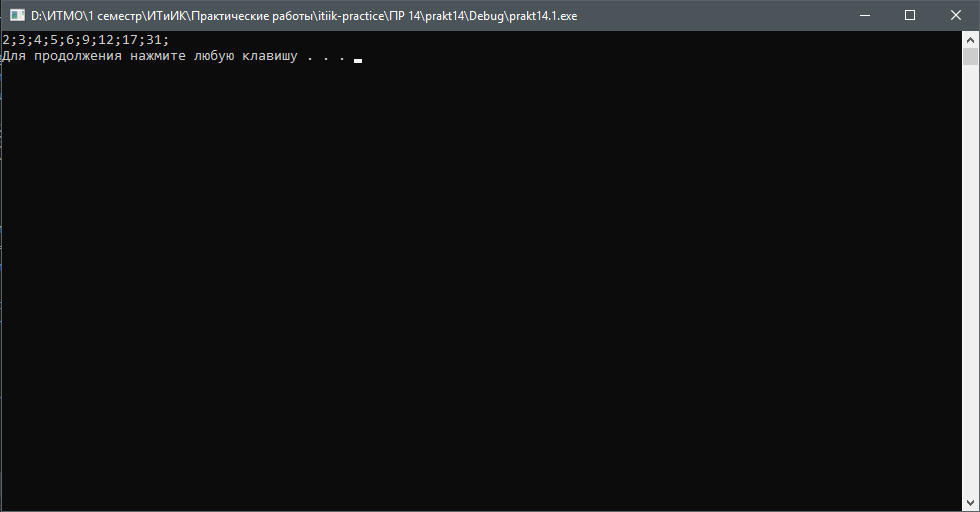


Рисунок 1 – сортировка массива

1. На основе функции сортировки был создан шаблон функции

template<class T>

void sorting(T arr[], int size) {

int j = 0;

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

{

T x = arr[i];

for (j = i - 1; j >= 0 && x < arr[j]; j--)

{

arr[j + 1] = arr[j];

}

arr[j + 1] = x;

}

}

1. В методе main были созданы массивы с разными типами данных. Была произведена их сортировка

int arr[] = { 9,3,17,6,5,4,31,2,12 };

double arrd[] = { 2.1, 2.3,1.7,6.6,5.3,2.44,3.1,2.4,1.2 };

char arrc[] = "Hello, word";

int k1 = sizeof(arr) / sizeof(arr[0]);

int k2 = sizeof(arrd) / sizeof(arrd[0]);

int k3 = sizeof(arrc) / sizeof(arrc[0]) - 1;

sorting(arr, k1);

for (int i = 0; i < k1; i++) cout << arr[i] << ";";

cout << endl;

sorting(arrd, k2);

for (int i = 0; i < k2; i++) cout << arrd[i] << ";";

cout << endl;

sorting(arrc, k3);

for (int i = 0; i < k3; i++) cout << arrc[i] << ";";

cout << endl;

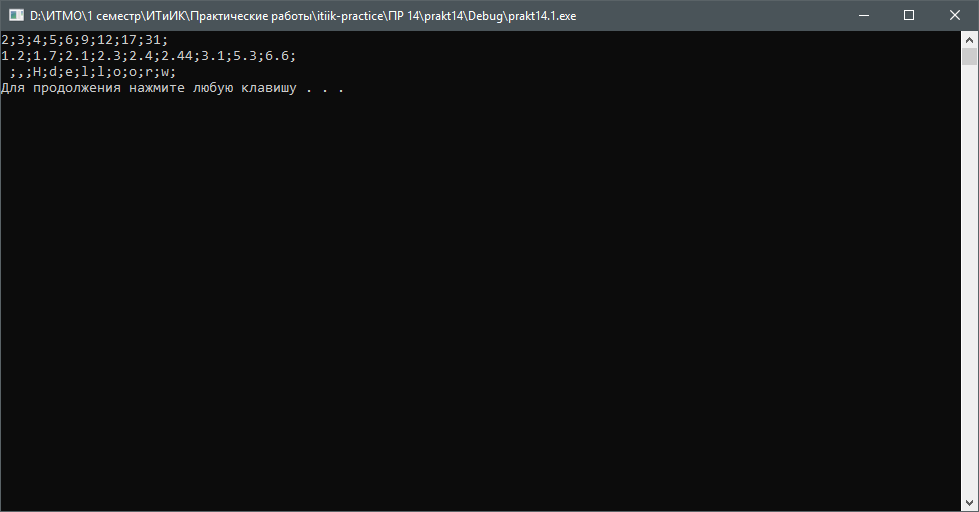


Рисунок 2 – сортировка массивов из разных типов данных

1. Был написан шаблон функции для вывода массивов в консоль

template<class T>

void printArray(T arr[], int size) {

cout << "\n";

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

{

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

}

cout << "\n";

}

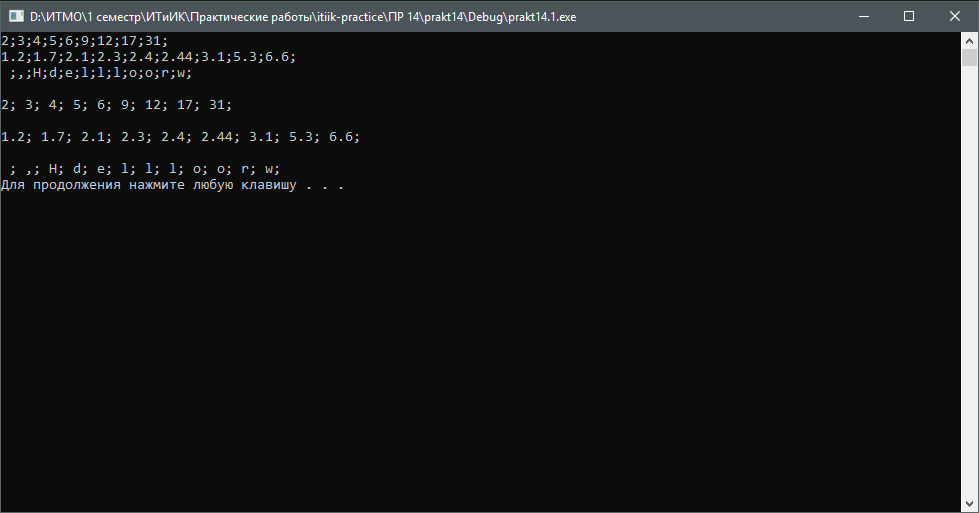


Рисунок 3 – вывод массивов в консоль

# Упражнение 2.

1. Был создан шаблон функции для подсчёта среднего арифметического массива любого типа данных

template<class T>

double srednArifm(T arr[], int size) {

T sum = 0;

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

{

sum += arr[i];

}

return (double)sum / size;

}

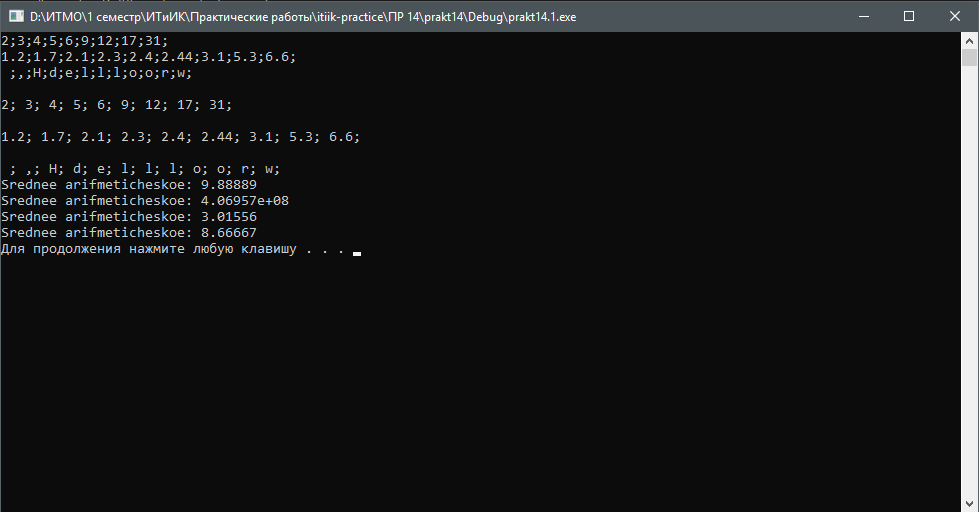


Рисунок 4 – среднее арифметическое массивов

# Упражнение 3.

1. Был создан новый проект
2. В нём был создан файл с исходным кодом

#include <iostream>

using namespace std;

template<class T, class T1> class arrayz

{

public:

arrayz(int size);

T1 sum();

T average\_value();

void show\_array();

int add\_value(T);

private:

T \*data;

int size;

int index;

};

template<class T, class T1> arrayz<T, T1>::arrayz(int size)

{

data = new T[size];

if (data == NULL)

{

cerr << "Error memory ---- exit program" << endl;

exit(1);

}

arrayz::size = size;

arrayz::index = 0;

}

template<class T, class T1> T1 arrayz<T, T1>::sum()

{

T1 sum = 0;

for (int i = 0; i < index; i++) sum += data[i];

return(sum);

}

template<class T, class T1> T arrayz<T, T1>::average\_value()

{

T1 sum = 0;

for (int i = 0; i < index; i++) sum += data[i];

return (sum / index);

}

template<class T, class T1> void arrayz<T, T1>::show\_array()

{

for (int i = 0; i < index; i++) cout << data[i] << ' ';

cout << endl;

}

template<class T, class T1> int arrayz<T, T1>::add\_value(T value)

{

if (index == size)

return(-1);

else

{

data[index] = value;

index++;

return(0);

}

}

int main()

{

arrayz<int, long> numbers(100);

arrayz<float, float> values(200);

int i;

for (i = 0; i < 50; i++) numbers.add\_value(i);

numbers.show\_array();

cout << "Sum = " << numbers.sum() << endl;

cout << "Average = " << numbers.average\_value() << endl;

for (i = 0; i < 100; i++) values.add\_value(i \* 100);

values.show\_array();

cout << "Sum = " << values.sum() << endl;

cout << "Average = " << values.average\_value() << endl;

system("pause");

return 0;

}

1. В данной программе реализован шаблон класса для работы с массивами различных значений

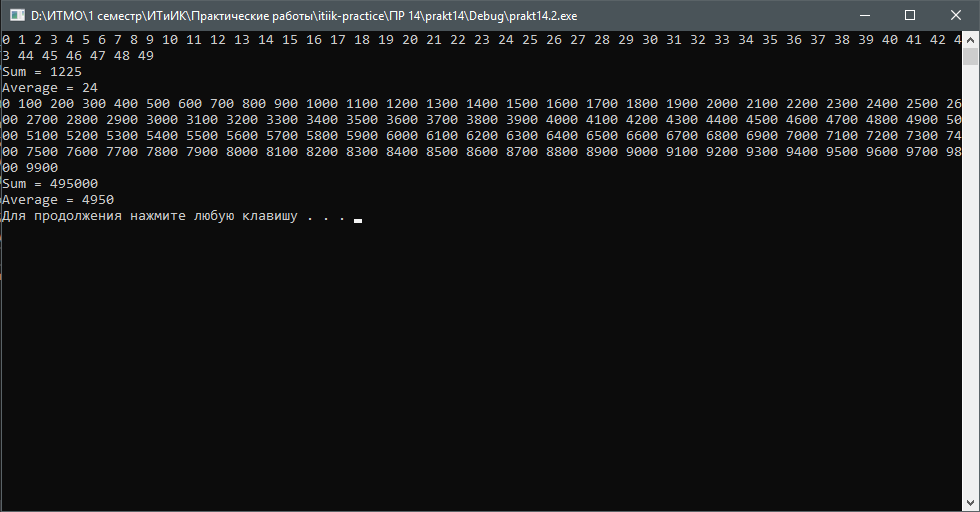


Рисунок 5 – программа работает исправно

# Вывод

В данной практической работе были рассмотрены возможности создания шаблонов функций и классов. Это позволяет передавать различные типы значений, не перегружая методы, которые их обрабатывают.

# Контрольные вопросы

1. Что такое шаблонные функции?

Шаблонная функция – функция, не привязанная к некоторым факторам (в нашем случае к типу данных).

1. Каким образом можно создать шаблонную функцию?

Объявление шаблонной функции происходит таким образом *template < параметры\_шаблона > описание\_функции.*