**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное образовательное бюджетное учреждение высшего образования**

**«Вологодский государственный университет»**

**(ВоГУ)**

* 1. **ОБЪЕКТНО-ОРЕНТИРОВАННОЕ программирование**
  2. **лабораторнАЯ РАБОТА №2**

**«Разработка классов на С++»**

**Направление подготовки: 09.03.01 Информатика и вычислительная техника**

**Направленность (профиль): Программное обеспечение средств вычислительной техники и автоматизированных систем**

**Форма обучения: очная**

**Институт: Математики, естественных и компьютерных наук**

**Кафедра: Автоматики и вычислительной техники**

**Группа: 4Б09 ПО-41**

**Студент: Махонин М.Н.**

**Руководитель: Сорокин А.Н.**

Вологда

2021 г.

**Цель работы:** научиться разрабатывать классы на языке С++.

**Вариант 2**

1. Разработать класс для работы с массивом чисел переменной длины. Класс должен содержать следующие элементы.

Поля (скрытые):

* количество элементов в массиве
* указатель на элементы массива

Методы:

* конструктор по умолчанию, формирующий массив из 10 элементов, заполненный нулями
* конструктор, формирующий массив из заданного количества элементов, заполненных нулями
* конструктор-копировщик
* деструктор
* метод для задания отдельных элементов массива
* методы для ввода элементов массива с клавиатуры и вывода на экран
* метод для удаления из массива последнего элемента (при этом нужно уменьшить на 1 размер массива и освободить память)
* метод для добавления в массив последнего элемента (при этом нужно увеличить на 1 размер массива и выделить память для добавляемого элемента)
* метод-функция для поиска требуемого элемента в массиве (метод возвращает -1, если элемент не найден, или индекс элемента в противном случае)

2. Реализовать для разработанного класса методы-операторы:

* + (плюс) - для соединения двух массивов, при этом длина нового массива должна равняться сумме длин исходных массивов; в новом массиве сначала располагаются все элементы первого массива, а затем все элементы второго массива.
* == (два знака равенства подряд) - для сравнения двух массивов, результатом этой операции должно быть любое ненулевое значение, если массивы имеют одинаковые размеры и все соответствующие элементы равны между собой; в противном случае результату присвоить значение 0.
* = (знак равенства) - присваивание

3. Реализовать класс в виде шаблона, позволяющего хранить данные различных типов.

4. Разработать программу, которая демонстрировала бы работоспособность всех методов данного класса.

#include <iostream>

using namespace std;

template <class T>

class Array

{

public:

Array();

Array(int n);

Array(Array& a);

void SetElement(int index, T el);

void KeyboardInput();

void PrintArray();

void Pop();

void Push(T el);

int IndexOf(T el);

Array  operator+ (Array& a);

int operator== (const Array& a);

Array<T>& operator = (const Array<T>& a);

~Array();

private:

int count;

T\* mas;

Array(T\* \_mas, int \_count);

};

template <class T>

Array<T>::Array(T\* \_mas, int \_count)

{

mas = \_mas;

count = \_count;

}

template <class T>

Array<T>::Array()

{

mas = new T[10];

count = 10;

for (int i = 0; i < 10; i++) {

mas[i] = 0;

}

}

template <class T>

Array<T>::Array(int n)

{

mas = new T[n];

count = n;

for (int i = 0; i < n; i++) {

mas[i] = 0;

}

}

template <class T>

Array<T>::Array(Array<T>& a)

{

mas = new T[a.count];

for (int i = 0; i < a.count; i++) {

mas[i] = a.mas[i];

}

count = a.count;

}

template<class T>

void Array<T>::SetElement(int index, T el)

{

if (index > count - 1) {

cout << "Индекс вне границ массива" << endl;

}

else {

mas[index] = el;

}

}

template<class T>

void Array<T>::KeyboardInput()

{

cout << "Ввдите количество элементов" << endl;

int n;

cin >> n;

count = n;

mas = new T[count];

for (int i = 0; i < count; i++) {

cin >> mas[i];

}

}

template<class T>

void Array<T>::PrintArray()

{

for (int i = 0; i < count; i++) {

cout << mas[i] << ' ';

}

}

template<class T>

void Array<T>::Pop()

{

T\* buf = new T[count - 1];

for (int i = 0; i < count - 1; i++) {

buf[i] = mas[i];

}

count = count - 1;

delete mas;

mas = buf;

}

template<class T>

void Array<T>::Push(T el)

{

T\* buf = new T[count + 1];

for (int i = 0; i < count; i++) {

buf[i] = mas[i];

}

buf[count] = el;

count = count + 1;

delete mas;

mas = buf;

}

template<class T>

int Array<T>::IndexOf(T el)

{

for (int i = 0; i < count; i++) {

if (el == mas[i]) {

return i;

}

}

return -1;

}

template<class T>

Array<T> Array<T>::operator+(Array& a)

{

T\* buf = new T[count + a.count];

for (int i = 0; i < count; i++) {

buf[i] = mas[i];

}

for (int i = 0; i < a.count; i++) {

buf[count + i] = a.mas[i];

}

Array<T> result(buf, count + a.count);

return result;

}

template<class T>

int Array<T>::operator==(const Array& a)

{

if (count != a.count) {

return 0;

}

for (int i = 0; i < count; i++) {

if (mas[i] != a.mas[i]) {

return 0;

}

}

return 1;

}

template<class T>

Array<T>& Array<T>::operator=(const Array<T>& a)

{

if (\*this == a) {

return \*this;

}

delete[] mas;

mas = new T[a.count];

for (int i = 0; i < a.count; i++) {

mas[i] = a.mas[i];

}

count = a.count;

return \*this;

}

template <class T>

Array<T>::~Array()

{

delete mas;

}

void main() {

setlocale(LC\_ALL, "Russian");

Array<int> a;

cout << "Вывод массива a" << endl;

a.PrintArray();

cout << endl << "Ввод вручную" << endl;

a.KeyboardInput();

cout << "Вывод массива a" << endl;

a.PrintArray();

Array<int> b;

for (int i = 0; i < 10; i++) {

b.SetElement(i, rand());

}

cout << endl << "Вывод массива b" << endl;

b.PrintArray();

cout << endl;

if (a == b) {

cout << "Массивы равны";

}

else {

cout << "Массивы неравны" << endl;

}

cout << "Вывод суммы" << endl;

(a + b).PrintArray();

cout << endl;

Array<int> c;

c = a + b;

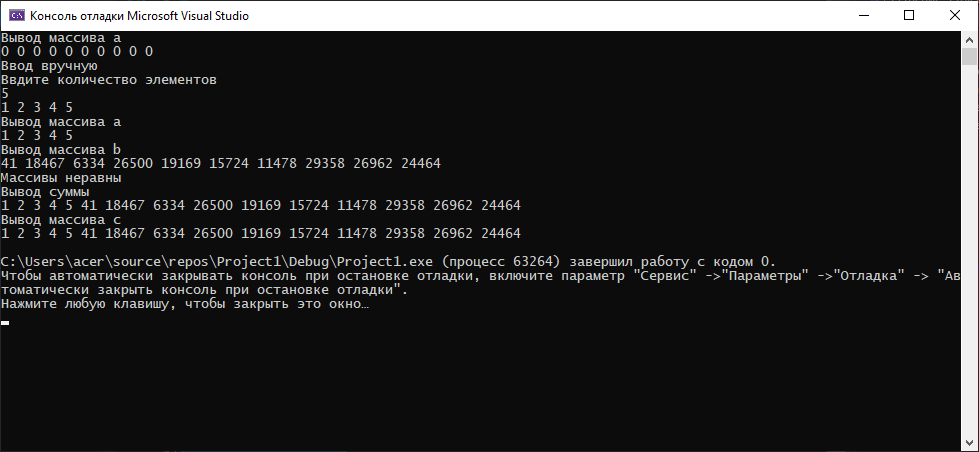
cout << "Вывод массива с" << endl;

c.PrintArray();

cout << endl;

}

Вывод:



**1. Что такое «константные методы»? Какие методы разработанного класса можно сделать константными?**

Константный метод — это метод, который гарантирует, что не будет изменять объект или вызывать неконстантные методы класса (поскольку они могут изменить объект).

Константные методы в моем классе:

* PrintArray
* Оператор суммы

Так как данные методы не изменяют объект класса, который их вызывает.

**2. Что такое «статическое поле» и «статический метод» класса? Для чего применяются статические свойства класса?**

Статический метод (static) класса – метод, не имеющий доступа к состоянию (полям) объекта, то есть к переменной this. Аналогично тому, как статический метод не привязан к объекту, статическое поле не является частью никакого объекта, оно будет создаваться в одном экземпляре и будет доступно из любого объекта класса или через имя класса.

**3.Что такое параметр типа для шаблона класса?**

Параметр-тип шаблона состоит из ключевого слова class или typename (в списке параметров они эквивалентны), за которым следует идентификатор. Оба ключевых слова обозначают, что последующее имя параметра относится к встроенному или определенному пользователем типу.