МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

КАФЕДРА ИИТ

ОТЧЁТ

по лабораторной работе №5

**«ПЕРЕГРУЗКА ОПЕРАЦИЙ»**

Выполнил:

студент 2 курса

группы ПО-9

Мисиюк Алексей Сергеевич

Проверил:

Козик И. Д.

**Порядок выполнения работы.**

1. Выбрать класс абстрактного типа данных (АТД) в соответствии с вариантом.

2. Определить и реализовать в классе конструкторы, деструктор, функции Input (ввод с клавиатуры) и Print (вывод на экран), перегрузить операцию присваивания.

3. Написать программу тестирования класса и выполнить тестирование.

4. Дополнить определение класса заданными перегруженными операциями ( в соответствии с вариантом).

5. Реализовать эти операции. Выполнить тестирование.

**Вариант 15**

АДТ − одномерный массив (вектор) вещественных чисел. Дополнительно перегрузить следующие операции:

+ − сложение векторов (a[i]+b[i] для всех i);

[] − доступ по индексу;

+ − добавить число к вектору (double+vector).

**Код программы**

#include <iostream>

using namespace std;

class Vector {

private:

double\* data; // указатель на массив данных

int size; // размер вектора

public:

// Конструктор по умолчанию

Vector() {

size = 0;

data = nullptr;

}

// Конструктор с параметром для создания вектора заданного размера

Vector(int n) {

size = n;

data = new double[size];

}

// Конструктор копирования

Vector(const Vector& other) {

size = other.size;

data = new double[size];

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

data[i] = other.data[i];

}

}

// Деструктор

~Vector() {

delete[] data;

}

// Функция ввода вектора с клавиатуры

void Input() {

cout << "Введите " << size << " элементов вектора: ";

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

cin >> data[i];

}

}

// Функция вывода вектора на экран

void Print() {

cout << "[";

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

cout << data[i];

if (i < size - 1) {

cout << ", ";

}

}

cout << "]" << endl;

}

// Перегрузка операции присваивания

Vector& operator=(const Vector& other) {

if (this != &other) {

delete[] data;

size = other.size;

data = new double[size];

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

data[i] = other.data[i];

}

}

return \*this;

}

// Перегрузка операции сложения векторов

Vector operator+(const Vector& other) {

Vector result(size);

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

result.data[i] = data[i] + other.data[i];

}

return result;

}

// Перегрузка операции доступа по индексу

double& operator[](int index) {

return data[index];

}

// Перегрузка операции добавления числа к вектору

Vector operator+(double value) {

Vector result(size);

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

result.data[i] = data[i] + value;

}

return result;

}

};

int main() {

Vector v1(3);

cout << "Введите элементы первого вектора:\n";

v1.Input();

Vector v2(2);

cout << "Введите элементы второго вектора:\n";

v2.Input();

Vector v3 = v1 + v2;

cout << "Результат сложения векторов: ";

v3.Print();

int index;

cout << "Введите индекс элемента для доступа: ";

cin >> index;

cout << "Элемент с индексом " << index << ": " << v3[index] << endl;

double value;

cout << "Введите число для добавления к вектору: ";

cin >> value;

Vector v4 = v3 + value;

cout << "Результат добавления числа к вектору: ";

v4.Print();

return 0;

}

**Результат:**