

Министерство науки и высшего образования Российской Федерации  
 Федеральное государственное бюджетное образовательное учреждение высшего образования  
«Московский государственный технический университет   
имени Н. Э. Баумана   
(национальный исследовательский университет)»   
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»  
КАФЕДРА «Информационная безопасность»

ОТЧЕТ  
по лабораторной работе № 2  
по учебной дисциплине «Алгоритмические языки»  
на тему: «Изучение перегрузки стандартных операций в языке Си++»

Вариант 4

Выполнил:   
Студент 1 курса, гр. ИУ8-24  
Дудко Егор

2024

1.Цель и задачи работы:

Овладение навыками разработки программ на языке Си++, использующих перегрузку стандартных операций. Для достижения цели необходимо выполнить следующие задачи:

- изучить необходимые учебные материалы, посвященные перегрузке стандартных операций в языке Си++ ;

- разработать программу на языке Си++ для решения заданного варианта задания;

- отладить программы;

- выполнить решение контрольного примера с помощью программы и ручной расчет контрольного примера;

- подготовить отчет по лабораторной работе.

2.Условие задачи:

Дан класс (например, с именем *Vector*), задающий вектор размерности *n*. Поля класса: указатель на массив, задающий вектор (тип элемента *double*), массив должен создаваться динамически; число элементов (размерность) вектора (тип *int*). Класс включает: конструктор без параметров, задающий пустой вектор (число элементов равно 0); конструктор, создающий объект вектор на основе обычного одномерного массива размерности *n*; конструктор копирования, конструктор перемещения, деструктор.

Необходимо перегрузить операции и продемонстрировать их работу. Перегрузить операцию *[]* (обращение к элементу вектора по индексу), операцию *=* (копирование вектора или создание копии вектора), операцию *\** (умножение числа на вектора), на выходе вектор такой же размерности, каждый элемент которого равен произведению соответствующего элемента исходного вектора на число.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| + сложение векторов, на выходе вектор, длина которого сумме длин векторов вначале идут элементы первого вектора, затем второго, если один из векторов задан обычным массивом, то считать, что его длина равна длине вектора, заданным объектом класса | double | Vector | Vector | Vector |

3.Код:

#include <iostream>

#include <fstream>

#include <vector>

**class** **Vector** { // Класс- вектор

**double**\* elements; // Указатель на массив (вектор)

**int** size; // Размерность вектора (число элементов) массива

**public:**

Vector() : elements(nullptr), size(**0**) {}

// Коструктор на входе массив, задающий вектор

Vector(**int** n) : size(n) {

elements = **new** **double**[size];

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

elements[i] = **0.0**;

}

Vector(**const** **double**\* arr, **int** n) : size(n) {

elements = **new** **double**[size];

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

elements[i] = arr[i];

}

Vector(**const** Vector& sec) : size(sec.size) {

elements = **new** **double**[size];

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

elements[i] = sec.elements[i];

}

Vector(Vector&& sec)

: elements(sec.elements), size(sec.size) {

sec.elements = nullptr;

sec.size = **0**;

}

~Vector() { // Деструктор

**delete**[] elements;

}

**double**& **operator**[](**int** index) { //Оператор- функция(перегрузка операции)

// обращения к элементу)

**return** elements[index];

}

Vector& **operator**=(**const** Vector& sec) { // Оператор- функция копирования объекта

**if** (**this** != &sec) {

**delete**[] elements;

size = sec.size;

elements = **new** **double**[size];

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

elements[i] = sec.elements[i];

}

**return** \***this**;

}

Vector& **operator**=(Vector&& sec) { // Оператор- функция перемещения объекта

**if** (**this** != &sec) {

**delete**[] elements;

elements = sec.elements;

size = sec.size;

sec.elements = nullptr;

sec.size = **0**;

}

**return** \***this**;

}

// Перегрузка ввывода

**friend** std::ostream& **operator**<<(std::ostream& out, **const** Vector& vec) {

**for** (**int** i = **0**; i < vec.size; ++i)

out << vec.elements[i] << " ";

**return** out;

}

// Перегрузка ввода

**friend** std::istream& **operator**>>(std::istream& in, Vector& vec) {

**for** (**int** i = **0**; i < vec.size; ++i)

in >> vec.elements[i];

**return** in;

}

// Оператор- функция вне класса

**friend** Vector **operator**+(**const** Vector& l, **const** Vector& r) {

Vector result(l.size + r.size);

**for** (**int** i = **0**; i < l.size; ++i)

result[i] = l.elements[i];

**for** (**int** i = **0**; i < r.size; ++i)

result[i + l.size] = r.elements[i];

**return** result;

}

};

**int** **main**() {

std::ifstream inputFile("input.txt");

**if** (!inputFile.is\_open()) {

std::cout << "Failed to open input file." << std::endl;

**return** **1**;

}

**int** n1, n2;

inputFile >> n1 >> n2;

Vector vec1(n1), vec2(n2);

inputFile >> vec1 >> vec2;

inputFile.close();

Vector sum = vec1 + vec2;

std::ofstream outputFile("output.txt");

**if** (!outputFile.is\_open()) {

std::cout << "Failed to open output file." << std::endl;

**return** **1**;

}

outputFile << "Vector 1: " << vec1 << std::endl;

outputFile << "Vector 2: " << vec2 << std::endl;

outputFile << "Sum: " << sum << std::endl;

outputFile.close();

**return** **0**;

}

4.Файл со входными данными – **input.txt:**

3 4

1.5 2.5 3.5

4.5 5.5 6.5 7.5

5.Файл с выходными данными– **output.txt:**

Vector 1: 1.5 2.5 3.5

Vector 2: 4.5 5.5 6.5 7.5

Sum: 1.5 2.5 3.5 4.5 5.5 6.5 7.5

6.Вывод:

В ходе выполнения лабораторной работы были изучены основы перегрузки стандартных операций в языке C++. Были реализованы операции работы с векторами.