Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)

**Лабораторная работа № 2. Изучение перегрузки стандартных операций в языке Си++**

Работу выполнил:

Алексеев Климентий Андреевич

ИУ8-24

Вариант 1

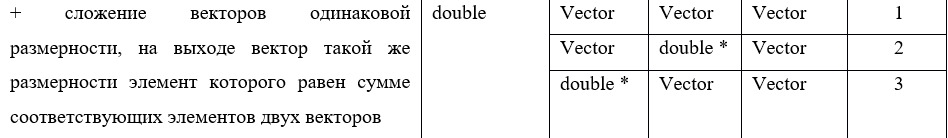
**Цель**

Цель работы состоит в овладении навыками разработки программ на языке Си++, использующих перегрузку стандартных операций.

**Условие задачи:**

Дан класс (например, с именем *Vector*), задающий вектор размерности *n*. Поля класса: указатель на массив, задающий вектор (тип элемента *int* или *double* в зависимости от варианта), 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 = (присваивание с копированием), операцию = (присваивание с перемещением), а также операцию вставки (<<) объекта в поток cout или в файл (объект класса ostream) и операцию извлечения (>>) объекта из потока cin или из файла (объект класса istream).



**Текст программы**

#include <iostream>

#include <fstream>

using namespace std;

class Vector {

int n;

double\* p;

public:

Vector(double\* p, int n)

{

this->n = n;

this->p = new double[n];

for (int i = 0;i < n;++i) this->p[i] = p[i];

}

Vector(int n) : n(n) {

cout << "test 1" << endl;

p = new double[n];

}

Vector(const Vector& other)

{

this->n = other.n;

this->p = new double[other.n];

for (int i = 0;i < other.n;++i) this->p[i] = other.p[i];

}

Vector (Vector&& prev)

{

cout << "test" << endl;

//this->n = prev.n; prev.n = 0;

//this->p = prev.p; prev.p = nullptr;

swap(p, prev.p);

swap(n, prev.n);

}

void print() const

{

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

cout << p[i] << " ";

}

cout << endl;

}

double& operator[](int index)

{

return p[index];

}

Vector& operator = (const Vector& other)

{

this->n = other.n;

this->p = new double[other.n];

for (int i = 0;i < other.n;++i) this->p[i] = other.p[i];

}

friend Vector operator \*(double x, Vector& v2);

friend Vector operator +(const Vector& v1, const Vector& v2);

~Vector()

{

if (this->p != nullptr) delete[] p;

}

friend ostream& operator<<(ostream& os, const Vector& v) { // Перегруженный оператор вывода в поток

for (int i = 0; i < v.n; i++) // Вывод значений через поток вывода

os << v.p[i] << " ";

return os;

}

friend istream& operator>>(istream& is, Vector& v) { // Перегруженный оператор ввода из потока

for (int i = 0; i < v.n; i++) // Считывание значений из потока ввода

is >> v.p[i];

return is;

}

};

Vector operator \*(double x, Vector& v)

{

Vector Vproiz(v.n);

for (int i = 0; i < v.n;++i) Vproiz[i] = v.p[i] \* x;

return Vproiz;

}

Vector operator +(const Vector& v1, const Vector& v2)

{

if (v1.n == v2.n) {

Vector Vsum(v1.n);

for (int i = 0; i < v1.n;++i) Vsum.p[i] = v1.p[i] + v2.p[i];

return Vsum;

}

else {

cout << "Diff size of mas";

}

}

Vector func() {

Vector vec(5);

return vec;

}

int main() {

setlocale(LC\_ALL, "RU");

ifstream inputFile("input.txt");

int size; double ch = 3;

inputFile >> size;

double\* mas = new double[size];

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

inputFile >> mas[i];

}

Vector V1(mas, size);

cout << "Vector 1: " << V1 << endl;

Vector V2 = ch \* V1;

cout << "Vector 2: " << V2 << endl;

Vector V3 = V1 + V2;

cout << "Vector 3: " << V3;

return 0;

}

Вывод программы: 4 8 12 16 20

Ручные рассчеты

Вектор V1 = {1 2 3 4 5}

Вектор V2 = {3 6 9 12 15}

1+3=4

2+6=8

3+9=12

4+12=16

5+15=20

Вектор V3 = {4 8 12 16 20}

Расчёты сходятся

**Вывод**

В ходе выполнения лабораторной работы были изучены и применены основы перегрузки операций в языке C++. Был разработан класс Vector для работы с векторами и перегружены операции --, [], =. Эти операции позволяют удобно работать с векторами, выполняя над ними математические операции и копируя/перемещая их значения. Полученные результаты демонстрируют успешную реализацию перегруженных операций и их корректную работу.