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:

Кадочников Вячеслав Павлович

ИУ8-24

Вариант 7

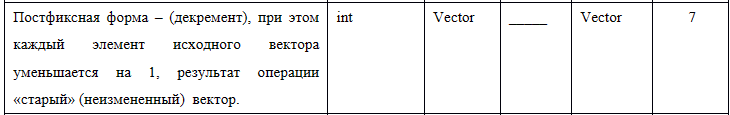
**Цель**

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>

using namespace std;

class Vector {

int\* p = nullptr; // Указатель на массив, задающий вектор

int n = 0; // Размерность вектора (число элементов)

public:

// Конструктор, создающий объект вектор на основе обычного одномерного массива размерности n

Vector(int\* p, int n) {

this->n = n;

this->p = new int[n]; // Выделение памяти под вектор

for (int i = 0; i < n; i++) this->p[i] = p[i]; // Копирование значений из переданного массива

}

// Конструктор, создающий пустой объект вектор заданной размерности

Vector(int n) : n(n) {

p = new int[n]; // Выделение памяти под вектор

}

// Конструктор копирования

Vector(const Vector& V) {

n = V.n;

p = new int[n];

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

p[i] = V.p[i];

}

// Конструктор перемещения

Vector(Vector&& V) {

std::swap(p, V.p); // Обмен указателями на массивы

std::swap(n, V.n); // Обмен значениями размерностей

}

// Функция печати вектора

void print() const {

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

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

cout << endl;

}

// Перегрузка оператора доступа к элементу вектора по индексу

int& operator[](int index) {

return p[index];

}

// Перегрузка оператора присваивания с копированием

Vector& operator =(const Vector& v2) {

if (this != &v2) {

n = v2.n;

if (p != nullptr) delete[] p; // Освобождаем память старого вектора

p = new int[n]; // Выделяем память для нового вектора

for (int i = 0; i < n; i++) p[i] = v2.p[i]; // Копируем значения из другого вектора

}

return \*this; // Возвращаем ссылку на текущий объект

}

// Перегрузка оператора присваивания с перемещением

Vector& operator =(Vector&& v2) {

if (this != &v2) {

std::swap(p, v2.p); // Обмен указателями на массивы

std::swap(n, v2.n); // Обмен значениями размерностей

}

return \*this; // Возвращаем ссылку на текущий объект

}

// Перегрузка оператора уменьшения каждого элемента вектора на 1

Vector operator--(int) {

Vector result(n); // Создаем новый объект вектора

for (int i = 0; i < n; i++) result.p[i] = p[i]--; // Заполняем новый вектор уменьшенными значениями

return result; // Возвращаем новый вектор

}

// Деструктор

~Vector() {

if (p != nullptr) delete[] p; // Освобождаем память, если она была выделена

}

};

int main() {

int m1[] = { 10, 5, 3, 4, 7 };

Vector V1(m1, 5);

Vector V2 = V1--;

V2.print();

return 0;

}

Вывод программы: 9 4 2 3 6

Ручные рассчеты

Исходный вектор {10, 5, 3, 4, 7}

10-1=9

5-1=4

3-1=2

4-1=3

7-1=6

Конечный вектор {9, 4, 2, 3, 6}

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.