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 23

Выполнил:   
Студент 1 курса, гр. ИУ8-24  
Спиридонов Олег

**Цель работы:**

Овладение навыками разработки программ на языке C++, использующих перегрузку стандартных операций.

**Задачи работы:**

* Изучить необходимые учебные материалы, посвященные перегрузке стандартных операций в языке Си++
* Разработать программу на языке Си++ для решения заданного варианта задания
* Отладить программу
* Выполнить решение контрольного примера с помощью программы и ручной расчет контрольного примера
* Подготовить отчет по лабораторной работе

**Условие задачи:**

1. **Описание класса Vector:**
   * Класс Vector задает вектор размерности n.
   * Поля класса:
     1. Число элементов (размерность) вектора.
     2. Указатель на массив типа int, задающий вектор.
2. **Конструкторы класса Vector:**
   * Конструктор без параметров, задающий пустой вектор (число элементов равно 0).
   * Конструктор, создающий объект вектора на основе обычного одномерного массива размерности n.
   * Конструктор копирования.
   * Конструктор перемещения.
   * Деструктор.
3. **Необходимые операции для перегрузки:**
   * Операция []: обращение к элементу вектора по индексу.
   * Операция = (присваивание с копированием).
   * Операция = (присваивание с перемещением).
   * Операция < (сравнение векторов (массивов)).
   * Операция вставки (<<) объекта в поток (объект класса ostream).
   * Операция извлечения (>>) объекта из потока (объект класса istream).
4. **Демонстрация разницы между конструктором копирования и конструктором перемещения.**
5. **Обработка входных и выходных данных:**
   * Все входные данные читаются из текстового файла input.txt.
   * Результаты выводятся в файл output.txt.

**Выполнение работы:**

Файл **input.txt:**

5

1 2 3 4 5

Файл **AL Laba 2 (operation overload).cpp**:

#include <iostream>

#include <fstream>

#include <vector>

using namespace std;

// ВАРИАНТ 23

class Vector {

friend ostream& operator<<(ostream& os, Vector& Vector);

friend istream& operator>>(istream& is, Vector& Vector);

friend bool operator<(Vector Vector, int\* massiv);

public:

Vector();

Vector(double\* massiv, int size);

Vector(Vector& Vector2);

Vector(Vector&& Vector2);

~Vector() {

if (p != nullptr) {

delete[] p; // Зачем []?

}

}

int operator[](int index) {

return p[index];

}

Vector operator=(const Vector& Vector2) {

n = Vector2.n;

if (p == nullptr) {

p = new int[n];

}

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

p[i] = Vector2.p[i]; // Почему нельзя Vector2[i]?

}

}

private:

int n = 0;

int\* p = nullptr;

};

Vector::Vector() {

};

Vector::Vector(double\* massiv, int size) {

this->n = size;

this->p = new int[n];

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

this->p[i] = massiv[i];

}

};

Vector::Vector(Vector& Vector2) {

this->n = Vector2.n;

this->p = new int[n];

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

this->p[i] = Vector2[i];

}

};

Vector::Vector(Vector&& Vector2) { // noexcept?

this->n = Vector2.n;

this->p = new int[n];

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

this->p[i] = Vector2[i];

}

Vector2.n = 0;

delete[] Vector2.p;

Vector2.p = nullptr;

};

ostream& operator<<(ostream& os, Vector& Vector) {

ofstream file("output.txt");

if (file.is\_open()) {

file << Vector.n << endl;

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

file << Vector.p[i] << ' ';

}

}else {

cout << "Output file didn't open." << endl;

}

os << Vector.n << endl;

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

os << Vector.p[i] << ' ';

}

return os;

}

istream& operator>>(istream& is, Vector& Vector) {

int amount;

int value;

vector <int> placeholder;

is >> amount;

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

is >> value;

placeholder.push\_back(value);

}

Vector.n = placeholder.size();

Vector.p = new int[Vector.n];

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

Vector.p[i] = placeholder[i];

}

return is;

}

bool operator<(Vector Vector, int\* massiv) {

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

if (Vector[i] < massiv[i]) {

return true;

}else if (Vector[i] > massiv[i]) {

return false;

}

}

return false;

}

int main() {

ofstream output\_file("output.txt");

output\_file.close();

ifstream input\_file("input.txt");

if (!input\_file.is\_open()) {

cout << "Input file didn't open." << endl;

}

Vector vector;

input\_file >> vector;

int massiv[5] = { 5, 4, 3, 2, 1 };

/\*cout << vector << endl;

for (int i = 0; i < 5; ++i) {

cout << massiv[i];

};

cout << endl;\*/

cout << (vector<massiv);

}

**Пример и контрольный расчет примера:**

Пусть заданы вектор длины 5 с элементами 1 2 3 4 5 и массив длины 5 с элементами 5 4 3 2 1. Проведя операцию сравнения, мы получим результат 1, следовательно, программа работает корректно.

**Вывод:**

В ходе выполнения лабораторной работы были изучены основы перегрузки стандартных операций в языке C++. Были реализованы операции работы с векторами, такие как обращение к элементу по индексу, копирование и перемещение векторов, сравнение векторов (массивов). В процессе работы были изучены способы перегрузки операций в классах C++, в том числе перегрузка как членами класса, так и дружественными функциями.