|  |  |
| --- | --- |
|  | **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, искусственный интеллект и системы управления” (ИУ) |

|  |  |
| --- | --- |
| КАФЕДРА | “Информационная безопасность” (ИУ8) |

Лабораторная работа № 2

ПО КУРСУ

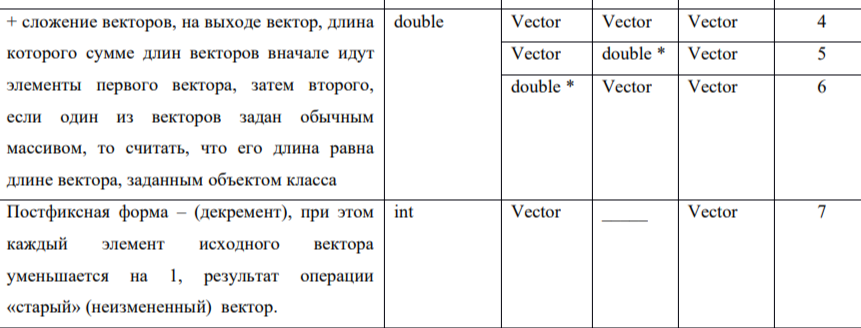
«Алгоритмические языки»

на тему «Изучение перегрузки стандартных операций в языке Си++»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ8-24 |  |  |  | А.А. Иванова |
|  | (Группа) |  |  |  | (И. О. Фамилия) |
|  |  |  |  |  |  |
| Преподаватель: |  |  |  |  | Д.В. Барыкин |
|  |  |  |  |  | (И.О. Фамилия) |

**Цель работы:**

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



**Код программы:**

1) *Заголовочный файл класса:*

#pragma once

#ifndef vector\_h

#define vector\_h

#include <iostream>

class Vector {

private:

double\* array;

int size;

public:

Vector();

Vector(int n);

Vector(const double\* arr, int n);

Vector(const Vector& other);

Vector(Vector&& other);

~Vector();

double& operator[](int index);

Vector& operator=(const Vector& other);

Vector& operator=(Vector&& other);

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

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

Vector operator+(const Vector& other) const;

};

#endif

2) *Файл реализации класса:*

#include "Vector.h"

#include <iostream>

#include <fstream>

Vector::Vector() : array(0), size(0) {}

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

array = new double[size];

}

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

array = new double[size];

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

array[i] = arr[i];

}

}

//Этот конструктор копирует все элементы из другого вектора в новый вектор

Vector::Vector(const Vector& other) : size(other.size) {

array = new double[size];

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

array[i] = other.array[i];

}

}

Vector::Vector(Vector&& other) : array(other.array), size(other.size) {

other.array = nullptr;

other.size = 0;

}

// деструктор

Vector::~Vector() {

delete[] array;

}

//Этот оператор возвращает ссылку на элемент массива по индексу index

double& Vector::operator[](int index) {

return array[index];

}

//оператор присваивания копирует все элементы из другого вектора в текущий вектор

Vector& Vector::operator=(const Vector& other) {

if (this != &other) {

delete[] array;

size = other.size;

array = new double[size];

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

array[i] = other.array[i];

}

}

return \*this;

}

//перемещает другой вектор в текущий вектор

Vector& Vector::operator=(Vector&& other) {

if (this != &other) {

delete[] array;

array = other.array;

size = other.size;

other.array = nullptr;

other.size = 0;

}

return \*this;

}

//оператор вывода выводит вектор в поток вывода

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

out << "(";

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

out << vec.array[i];

if (i < vec.size - 1) out << ", ";

}

out << ")";

return out;

}

//оператор ввода считывает вектор из потока ввода

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

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

in >> vec.array[i];

}

return in;

}

//оператор сложения для векторов

Vector Vector::operator+(const Vector& other) const

{

Vector result(size + other.size);

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

result.array[i] = array[i];

}

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

result.array[size + i] = other.array[i];

}

return result;

}

void readVectorsFromFile(const std::string& filename, Vector& vec1, Vector& vec2) {

std::ifstream file(filename);

if (file.is\_open()) {

int size1, size2;

file >> size1;

vec1 = Vector(size1);

file >> vec1;

file >> size2;

vec2 = Vector(size2);

file >> vec2;

file.close();

}

else {

std::cerr << "Error opening file for reading." << std::endl;

}

}

void writeVectorToFile(const std::string& filename, const Vector& vec) {

std::ofstream file(filename);

if (file.is\_open()) {

file << vec;

file.close();

}

else {

std::cerr << "Error opening file for writing." << std::endl;

}

}

int main()

{

Vector vec1, vec2;

readVectorsFromFile("input.txt", vec1, vec2);

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

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

Vector sum = vec1 + vec2;

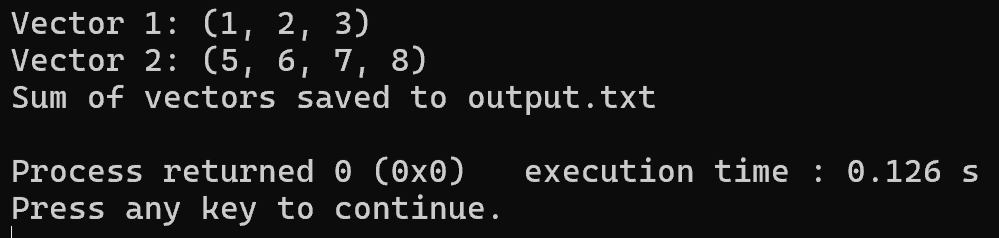
writeVectorToFile("output.txt", sum);

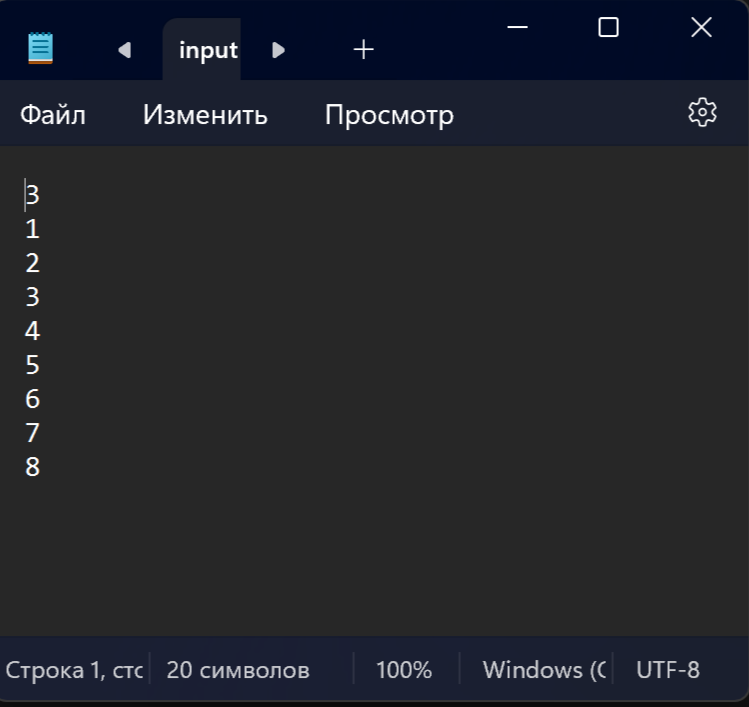
std::cout << "Sum of vectors saved to output.txt" << std::endl;

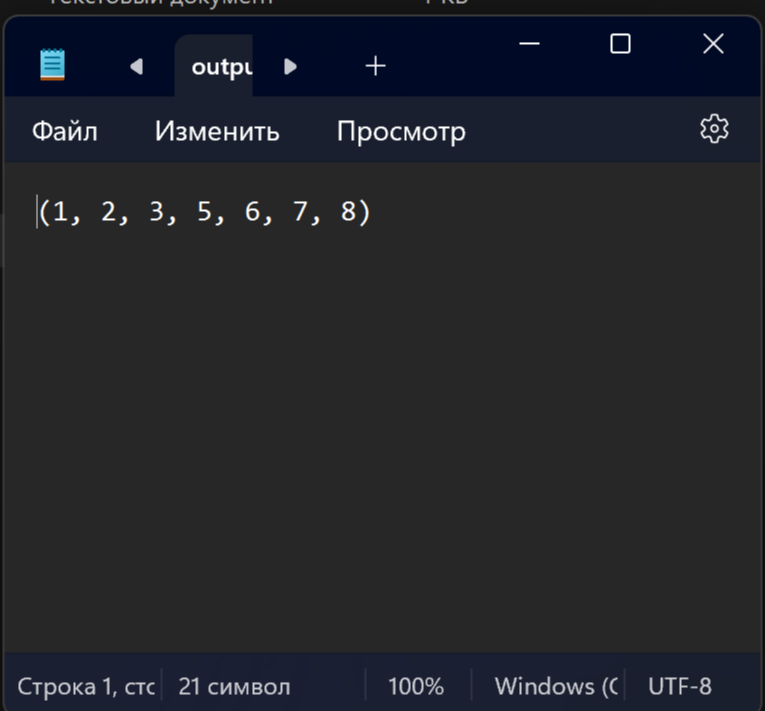
return 0;

}

**Результат работы программы:**







**Контрольный расчет:**

Подставив данные программы в калькулятор и пересчитав руками, я получила такой же результат.

**Выводы:**

Выполнив лабораторную работу №2, я узнала тему «Изучение перегрузки стандартных операций в языке Си++», основные понятия объектно-ориентированного программирования языка Си++ – классов и объектов, применила классы, объекты и перегрузки операций на практике.