|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 ИУ8 “КОМПЬЮТЕРНАЯ БЕЗОПАСНОСТЬ”

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

Студент Лустин Павел Евгеньевич

*(*Фамилия И.О)

Группа\_\_ИУ8-24\_\_\_\_\_\_\_\_

Студент  **\_\_\_\_\_\_08.03\_\_\_\_\_\_\_\_\_**  Лустин \_\_ \_\_

*дата (*Фамилия И.О)

Преподаватель **\_\_\_\_\_\_08.03\_\_\_\_\_\_\_**  Барыкин Дмитрий

*дата (*Фамилия И.О)*.*

*2024 г.*

*Цель работы:*

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

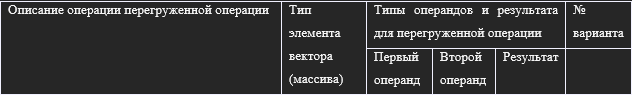
Условие задачи:

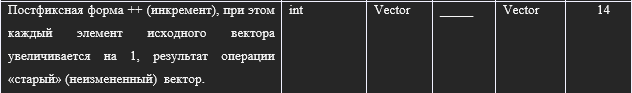
Дан класс (например, с именем *Vector*), задающий вектор размерности *n*. Поля класса: указатель на массив, задающий вектор (тип элемента *int* или *double* в зависимости от варианта), массив должен создаваться динамически, число элементов (размерность) вектора (тип *int*). Класс включает: конструктор 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). *Также продемонстрировать разницу между конструктором копирования и конструктором перемещения и между операциями присваивания с копированием и перемещением. Исходные коды класса разместить в двух файлах: в заголовочном файле класса и файле реализации класса.*

*При выполнении работы все входные данные читаются из текстового файла input.txt (создать этот файл любым текстовым редактором), результаты выводятся в файл output.txt. В отчете представить содержимое этих файлов.*

*Индивидуальное задание (Вариант 14)*

**

**

*Текст программы с комментариями*

*Заголовочный файл (header\_file.cpp):*

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <sstream>

*Файл реализации класса:*

#include "header\_file.cpp" //включение заголовочного файла

using namespace std;

class Vector //Класс Vector

{

int n;

int\* p;

public:

Vector() //Пустой конструктор

{

n = 0;

p = nullptr;

}

Vector(int n, int\*p) //Конструктор с начальными значениями, взятыми из передаваемого массива

{

this->n = n;

this->p = new int[n];

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

{

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

}

}

Vector(const Vector& V) //Конструктор копирования

{

this->n = V.n;

this->p = new int[n];

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

{

this->p[i] = V.p[i];

}

}

Vector(Vector&& V) //Конструктор перемещения

{

this->n = V.n;

this->p = V.p;

V.n = 0;

V.p = nullptr;

}

void print\_file(ofstream& fout) //Функция класса печати вектора

{

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

{

fout << p[i] << " ";

}

}

int & operator [](int i) //Перегруженная операция обращения к элементу вектора по индексу

{

return p[i];

}

Vector& operator = (const Vector& V) //Перегруженная операция присваивания с копированием

{

if (this != &V)

{

this->n = V.n;

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

p = new int[n];

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

{

this->p[i] = V.p[i];

}

}

return \*this;

}

friend ostream& operator << (ostream& fout, const Vector& V); //Описание дружественной перегруженной операции вставки объекта в поток cout

friend istream& operator >> (istream& fin, const Vector& V); //Описание дружественной перегруженной операции извлечения объекта из потока cin

friend Vector& operator ++ (Vector& V); //Описание дружественной перегруженной операции инкремента

Vector& operator = (Vector&& V) //Перегруженная операция присваивания с перемещением

{

if (this != &V)

{

this->n = V.n;

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

p = V.p;

V.p = nullptr;

}

return \*this;

}

~Vector() //Деструктор

{

if (p != nullptr) delete[] p;

}

};

ostream& operator << (ostream& fout, const Vector& V) //Объявление дружественной перегруженной операции вставки объекта в поток cout

{

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

{

fout << V.p[i] << " ";

}

return fout;

}

istream& operator >> (istream& fin, const Vector& V) //Объявление дружественной перегруженной операции извлечения объекта из потока cin

{

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

{

fin >> V.p[i];

}

return fin;

}

Vector & operator ++ (Vector& V) //Объявление дружественной перегруженной операции инкремента

{

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

{

V.p[i]++;

}

return V;

}

int main()

{

setlocale(LC\_ALL, "Rus"); //реализуем возможность печати русских букв

int count = 0, temp = 0, count2 = 0;

string str, strx;

vector <string> vecstr; //вектор из строк, в которые будут поочередно записаны все строчки файла (1 строка = 1 строчка файла)

vector <int> vecint; //вектор для временного хранения всех чисел n-го массива

ifstream fin;

fin.open("input.txt"); //открываем файл для чтения

while(!fin.eof()) //с начала до конца файла построчно записываем строки в массив

{

getline(fin, str);

vecstr.push\_back(str);

++count;

str = "";

}

fin.close(); //закрываем файл после чтения

int\*\* arr = new int\* [count]; //массив, в который будут попадать все массивы из цифр, полученные из файла

vector <int> countarr; //массив из чисел, являющихся размерностями массивов

for(int i = 0; i < count; ++i) //здесь мы из строк с наборами чисел получаем готовые массивы цифр и количества элементов в них

{

str = vecstr[i];

for(int j = 0; j < size(str); ++j) //цикл по выделению каждого отдельного числа из строки и записи этого числа в отдельную строку для последующего преобразования новой строки в число

{

if (str[j] == ' ') continue;

if(str[j] != ' ')

{

strx += str[j];

}

istringstream(strx) >> temp; //преобразование строки в число

vecint.push\_back(temp);

strx = ""; //очищаем строку для добавления в неё цифр, образующих следующее число массива

++count2;

}

arr[i] = new int[count2]; //объявляем один из n массивов, находящихся в файле

countarr.push\_back(count2);

for(int g = 0; g < count2; ++g) //цикл для придания значений всем элементам массива, объявленного в 145 строке

{

arr[i][g] = vecint[g];

}

vecint.clear(); //очищаем вектор для добавления в него новых чисел уже другого массива из файла

count2 = 0;

}

ofstream fout;

fout.open("output.txt"); //открываем файл для демонстрации работы всех перегруженных операций

Vector class\_vec1(countarr[0], arr[0]); //с помощью конструктора создаем класс Вектор1

fout << "Первый вектор: ";

class\_vec1.print\_file(fout);

fout << endl;

Vector class\_vec2(countarr[1], arr[1]); //с помощью конструктора создаем класс Вектор2

fout << "Второй вектор: ";

class\_vec2.print\_file(fout);

fout << endl << endl;

fout << "Операция копирования: ";

class\_vec1 = class\_vec2; //копируем Вектор2 в Вектор1 (получаем 2 одинаковых Вектора)

fout << "Первый вектор: ";

class\_vec1.print\_file(fout);

fout << endl;

fout << "Второй вектор: ";

class\_vec2.print\_file(fout);

fout << endl << endl;

fout << "Операция []: ";

fout << endl;

fout << "Третий элемент массива класса class\_vec1: " << class\_vec1[2]; //с помощью перегруженной операции [] обращаемся к 3-му элементу Вектора1

fout << endl << endl;

fout << "Операция вставки объекта в поток (<<) : ";

fout << class\_vec1; //при помощи перегруженной операции << вставляем Вектор1 в поток

fout << endl << endl;

fout << "Операция извлечения объекта из потока элементов (>>) : ";

cin >> class\_vec1; //при помощи перегруженной операции >> присваиваем новые значения Вектору1 (с клавиатуры)

fout << "Измененный вектор: ";

class\_vec1.print\_file(fout);

fout << endl << endl;

fout << "Операция постфиксной формы ++(инкремент) : ";

++class\_vec1; //при помощи перегруженной операции ++ (инкремента) каждому элементу Вектора1 прибавляем единицу

cout << endl;

fout << endl;

fout << "Измененный вектор: ";

class\_vec1.print\_file(fout);

fout.close(); //закрываем файл после записи

return 0;

}

Содержимое файла input.txt:

*1 2 3*

*4 5 6 7*

*Содержимое файла output.txt:*

*Первый вектор: 1 2 3*

*Второй вектор: 4 5 6 7*

*Операция копирования: Первый вектор: 4 5 6 7*

*Второй вектор: 4 5 6 7*

*Операция []:*

*Третий элемент массива класса class\_vec1: 6*

*Операция вставки объекта в поток (<<) : 4 5 6 7*

*Операция извлечения объекта из потока элементов (>>) : Измененный вектор: 1 2 3 4*

*Операция постфиксной формы ++(инкремент) :*

*Измененный вектор: 2 3 4 5*

*Вывод:*

*Я* овладел навыками разработки программ на языке Си++, использующих перегрузку стандартных операций и смог выполнить все задачи, продемонстрировать работу программы.