ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ ВЫСШЕГО

ПРОФЕССИОНАЛЬНОГО ОБРАЗОВНИЯ

«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра АСУ

ОТЧЕТ

по лабораторной работе № 1

по дисциплине

«Объектно-ориентированное программирование»

тема: «Обобщённое программирование. Шаблоны функций и классов»

Выполнил:

студент группы ИСТ-19а

Агарков И. А.

Проверили:

Поляков А.И.

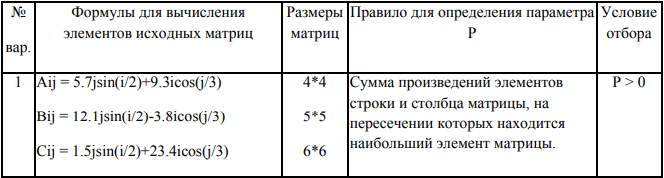
Донецк – 2021

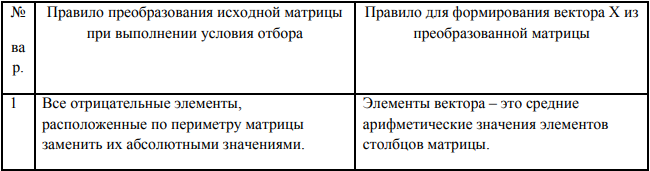
**Постановка задачи:**

При выполнении лабораторной работы требуется создать шаблонные классы для работы с векторами и матрицами, содержащими элементы произвольного типа. Выполнение лабораторной работы следует осуществлять с учетом следующих требований:

1. Определить шаблоны вектора и матрицы.
2. Определить конструкторы копирования.
3. Переопределить операции:
   1. вывода в стандартный поток ввода-вывода;
   2. индексации элементов;
4. Определить методы для:
   1. a. ввода с клавиатуры;
   2. заполнения случайными значениями;
   3. определения длины.
5. Создать шаблон функции, выполняющий обработку матрицы и формирования вектора в соответствии с индивидуальным заданием.
6. В лабораторной работе выполнить обработку нескольких векторов разных типов: char, int, long, double, long double. (Типы, несовместимые с индивидуальным заданием разрешается исключить).

**Индивидуальное задание:**

****

****

**Листинг программы:**

Файл Vector.h:

#pragma once

#include <iostream>

#include <iomanip>

#include "conio.h"

#include "Windows.h"

using namespace std;

template<typename T>

class Vector {

private:

int count;

T\* items;

public:

Vector() {

this->count = 0;

this->items = nullptr;

}

Vector(int count) {

this->count = count;

this->items = new T[count];

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

items[i] = 0;

}

Vector(const Vector<T>& orig) {

this->items = new T[orig.count];

this->count = orig.count;

for (int i = 0; i < this->count; i++)

this->items[i] = orig.items[i];

}

~Vector() {

delete[] this->items;

}

void setsize(int count) {

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

this->count = count;

this->items = new T[count];

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

this->items[i] = 0;

}

void fillrand() {

for (int i = 0; i < this->count; i++)

this->items[i] = (T)(0.1 \* (rand() % (201) - 100));

}

void fillmanual() {

for (int i = 0; i < this->count; i++)

cin >> this->items[i];

}

T& operator[] (int i) {

if (i >= 0 && i < this->count) return this->items[i];

else {

cout << "Index error! Aborting." << endl;

system("pause");

exit(0);

}

}

template <class T>

friend ostream& operator << (ostream&, const Vector<T>&);

};

template <class T>

ostream& operator << (ostream& out, const Vector<T>& vector) {

for (int i = 0; i < vector.count; i++)

out << fixed << setprecision(4) << setw(10) << vector.items[i] << " ";

return out;

}

Файл Matrix.h:

#pragma once

#include "Vector.h"

#include "math.h"

using namespace std;

template<typename T>

class Matrix {

private:

int cols, rows;

Vector<T>\* vectors;

public:

Matrix() {

this->vectors = nullptr;

this->rows = this->cols = 0;

}

Matrix(int cols, int rows) {

this->rows = rows;

this->cols = cols;

this->vectors = new Vector<T>[rows];

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

this->vectors[i].setsize(cols);

}

Matrix(Matrix& orig) {

this->rows = orig.rows;

this->cols = orig.cols;

this->vectors = new Vector<T>[this->rows];

for (int i = 0; i < this->rows; i++) {

this->vectors[i].setsize(this->cols);

for (int j = 0; j < this->cols; j++)

this->vectors[i][j] = orig.vectors[i][j];

}

}

Matrix(int cols, int rows, float A, float B) {

this->cols = cols;

this->rows = rows;

this->vectors = new Vector<T>[rows];

for (int i = 0; i < this->rows; i++) {

this->vectors[i].setsize(this->cols);

for (int j = 0; j < this->cols; j++)

this->vectors[i][j] = A \* (j + 1) \* sinf((i + 1) / 2.0) + B \* (i + 1) \* cosf((j + 1) / 3.0);

}

}

void findmax(int& imax, int& jmax) {

T max = (T)FLT\_MIN;

for (int i = 0; i < this->rows; i++) {

for (int j = 0; j < this->cols; j++) {

if (this->vectors[i][j] > max) {

max = this->vectors[i][j];

imax = i;

jmax = j;

}

}

}

}

T calculatep() {

T column = 1, row = 1;

int imax, jmax;

this->findmax(imax, jmax);

for (int i = 0; i < this->rows; i++) column \*= this->vectors[i][jmax];

for (int j = 0; j < this->cols; j++) row \*= this->vectors[imax][j];

return column + row;

}

void transform() {

for (int i = 0; i < this->rows; i++) {

this->vectors[i][0] = (T)fabs(this->vectors[i][0]);

this->vectors[i][this->cols - 1] = (T)fabs(this->vectors[i][this->cols - 1]);

}

for (int j = 0; j < this->cols; j++) {

this->vectors[0][j] = (T)fabs(this->vectors[0][j]);

this->vectors[this->rows - 1][j] = (T)fabs(this->vectors[this->rows - 1][j]);

}

}

T columnaverage(int col) {

if (col < 0 || col > this->cols) {

cout << "Index error! Aborting." << endl;

system("pause");

exit(0);

}

T average = 0;

for (int i = 0; i < this->rows; i++)

average += this->vectors[i][col];

return average / this->rows;

}

Vector<T> getvector() {

Vector<T> result(this->cols);

for (int i = 0; i < this->cols; i++)

result[i] = this->columnaverage(i);

return result;

}

void fillrand() {

for (int i = 0; i < this->rows; i++)

this->vectors[i].fillrand();

}

void fillmanual() {

for (int i = 0; i < this->rows; i++) {

cout << "Row #" << i + 1 << ": ";

this->vectors[i].fillmanual();

cout << endl;

}

}

int getrows() { return this->rows; }

int getcols() { return this->cols; }

Vector<T> operator[](int row) {

if (row >= 0 && row < this->rows) return this->vectors[row];

else {

cout << "Index error! Aborting." << endl;

system("pause");

exit(0);

}

}

template <class T>

friend ostream& operator << (ostream& out, Matrix<T>& matrix);

};

template <typename T>

ostream& operator << (ostream& out, Matrix<T>& matrix) {

for (int i = 0; i < matrix.getrows(); i++) {

out << matrix.vectors[i] << endl;

}

return out;

}

Файл main.cpp:

#include "Matrix.h"

int main() {

Matrix<int> A(4, 4, 5.7, 9.3);

Matrix<float> B(5, 5, 12.1, -3.8);

Matrix<double> C(6, 6, 1.5, 23.4);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

cout << "Matrix<int> A:" << endl << A << endl;

if (A.calculatep() > 0) {

cout << "P > 0 for matrix A, transforming." << endl << endl;

A.transform();

cout << "New Matrix<int> A:" << endl << A << endl;

}

Vector<int> A\_vector = A.getvector();

cout << "Vector A:" << endl << A\_vector << endl;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

cout << endl << "=======================================" << endl << endl;

cout << "Matrix<float> B:" << endl << B << endl;

if (B.calculatep() > 0) {

cout << "P > 0 for matrix B, transforming." << endl << endl;

B.transform();

cout << "New Matrix<int> B:" << endl << B << endl;

}

Vector<float> B\_vector = B.getvector();

cout << "Vector B:" << endl << B\_vector << endl;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

cout << endl << "=======================================" << endl << endl;

cout << "Matrix<double> C:" << endl << C << endl;

if (C.calculatep() > 0) {

cout << "P > 0 for matrix C, transforming." << endl << endl;

C.transform();

cout << "New Matrix<int> C:" << endl << C << endl;

}

Vector<double> C\_vector = C.getvector();

cout << "Vector C:" << endl << C\_vector << endl;

cout << endl;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

system("pause");

}

**Результат выполнения**:

Matrix<int> A:

11 12 13 13

22 24 24 23

32 33 32 29

40 39 35 29

P > 0 for matrix A, transforming.

New Matrix<int> A:

11 12 13 13

22 24 24 23

32 33 32 29

40 39 35 29

Vector A:

26 27 26 23

=======================================

Matrix<float> B:

2.2102 8.6157 15.3500 22.3103 29.3690

3.0001 14.3909 26.4391 38.9394 51.6365

1.2972 15.1803 30.0496 45.5970 61.4397

-3.3608 10.0595 24.7949 40.4344 56.4675

-10.7127 -0.4488 11.4588 24.4965 38.0263

P > 0 for matrix B, transforming.

New Matrix<int> B:

2.2102 8.6157 15.3500 22.3103 29.3690

3.0001 14.3909 26.4391 38.9394 51.6365

1.2972 15.1803 30.0496 45.5970 61.4397

3.3608 10.0595 24.7949 40.4344 56.4675

10.7127 0.4488 11.4588 24.4965 38.0263

Vector B:

4.1162 9.7390 21.6185 34.3555 47.3878

=======================================

Matrix<double> C:

22.8311 19.8280 14.8005 8.3811 1.3558 -5.4230

45.4862 39.3039 29.0728 16.0579 1.8312 -11.9024

67.8322 58.1618 42.4179 22.4986 0.7614 -20.2361

89.8119 76.2869 54.6641 27.4740 -2.1400 -30.7677

111.4577 93.7442 65.9085 31.1136 -6.7111 -43.3029

132.8836 110.7619 76.4935 33.8741 -12.3812 -57.1569

P > 0 for matrix C, transforming.

New Matrix<int> C:

22.8311 19.8280 14.8005 8.3811 1.3558 5.4230

45.4862 39.3039 29.0728 16.0579 1.8312 11.9024

67.8322 58.1618 42.4179 22.4986 0.7614 20.2361

89.8119 76.2869 54.6641 27.4740 -2.1400 30.7677

111.4577 93.7442 65.9085 31.1136 -6.7111 43.3029

132.8836 110.7619 76.4935 33.8741 12.3812 57.1569

Vector C:

78.3838 66.3478 47.2262 23.2332 1.2464 28.1315

Press any key to continue . . .