

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут
імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 4 з дисципліни
«Основи програмування - 2.
Модульне програмування»

«Перевантаження операторів»

Варіант 5

Виконав студент

ІП-15, Буяло Дмитро Олександрович
(шифр, прізвище, ім'я, по батькові)

Перевірив

Вечерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 4

Перевантаження операторів

Мета – вивчити механізми створення класів з використанням перевантажених операторів (операцій).

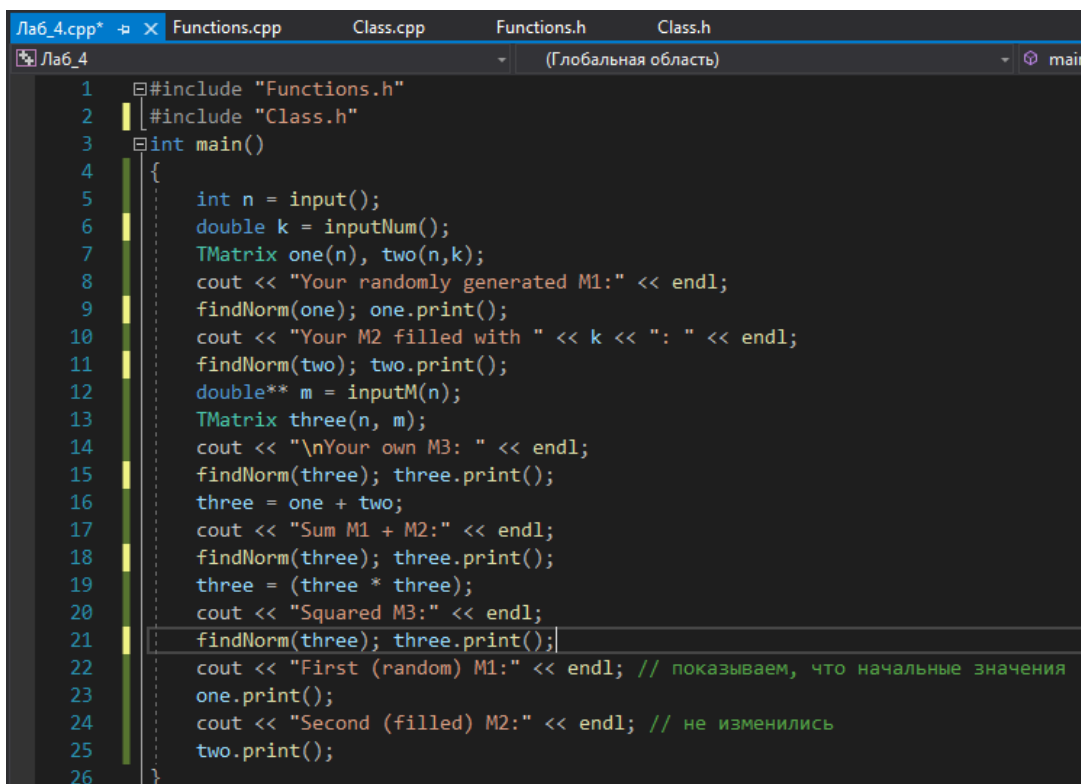
Індивідуальне завдання

Варіант 5

Завдання

Визначити клас «Квадратна матриця» розмірності n . Реалізувати для нього декілька конструкторів, геттери, метод обчислення норми матриці. Перевантажити оператори додавання «+» і множення «*» матриць. Створити три матриці (M1, M2, M3), використовуючи різні конструктори. Визначити матрицю M3 як суму матриць M1 та M2. Отриману матрицю M3 піднести до квадрату. Знайти норму нової матриці M3.

1. Код на C++



```
Лаб_4.cpp  Functions.cpp  Class.cpp  Functions.h  Class.h
Лаб_4  (Глобальная область)  main
1  #include "Functions.h"
2  #include "Class.h"
3  int main()
4  {
5      int n = input();
6      double k = inputNum();
7      TMatrix one(n), two(n,k);
8      cout << "Your randomly generated M1:" << endl;
9      findNorm(one); one.print();
10     cout << "Your M2 filled with " << k << ": " << endl;
11     findNorm(two); two.print();
12     double** m = inputM(n);
13     TMatrix three(n, m);
14     cout << "\nYour own M3: " << endl;
15     findNorm(three); three.print();
16     three = one + two;
17     cout << "Sum M1 + M2:" << endl;
18     findNorm(three); three.print();
19     three = (three * three);
20     cout << "Squared M3:" << endl;
21     findNorm(three); three.print();
22     cout << "First (random) M1:" << endl; // показываем, что начальные значения
23     one.print();
24     cout << "Second (filled) M2:" << endl; // не изменились
25     two.print();
26 }
```

```

Лаб_4.cpp  Functions.cpp*  Class.cpp  Functions.h  Class.h
Лаб_4      (Глобальная область)  invalidInput(int & n

1  #include "Functions.h"
2  #include "Class.h"
3
4
5  int input() { // вводим размерность матрицы
6      int n;
7      cout << "Enter the dimention of your M1: "; cin >> n;
8      invalidInput(n);
9      cout << endl;
10     return n;
11 }
12
13 void invalidInput(int& n) { // проверка на валидность ввода
14     while (!cin || n < 1) {
15         cin.clear();
16         cin.ignore(64, '\n');
17         cout << "Your input is wrong, try again: "; cin >> n;
18     }
19 }
20
21 double inputNum() { // вводим число, от которого начнем заполнение M2
22     double k;
23     cout << "Enter the number you want to fill M2 with: "; cin >> k;
24     invalidInput(k);
25     cout << endl;
26     return k;
27 }
28
29 void invalidInput(double& input) { // перегруженная функция проверки валидности для дабла
30     while (!cin) {
31         cin.clear();
32         cin.ignore(64, '\n');
33         cout << "Your input is wrong, try again: "; cin >> input;
34     }
35 }
36
37 void findNorm(TMatrix& obj) { // одной функцией вызываем все нормы
38     obj.findNormaM();
39     obj.findNormal();
40     obj.findNormaE();
41 }
42
43 double** inputM(int n) { // ввод M3 с клавиатуры
44     double** m = new double* [n];
45     for (int i = 0; i < n; i++) {
46         m[i] = new double[n];
47         for (int j = 0; j < n; j++) {
48             cout << "M3[" << i << "][" << j << "] = ";
49             cin >> m[i][j];
50         }
51     }
52     return m;
53 }

```

```

Лаб_4.cpp  Functions.cpp  Class.cpp  Functions.h  Class.h
Лаб_4  → TMatrix

1  #include "Class.h"
2
3
4  double TMatrix::getEl(int i, int j) {
5      return matrix[i][j];
6  }
7
8  double TMatrix::getNormaE() {
9      return normaEuclid;
10 }
11 void TMatrix::setNormaE(double euclid) {
12     normaEuclid = euclid;
13 }
14 double TMatrix::getNormaM() {
15     return normaM;
16 }
17 void TMatrix::setNormaM(double m) {
18     normaM = m;
19 }
20 double TMatrix::getNormal() {
21     return normal;
22 }
23 void TMatrix::setNormal(double l) {
24     normal = l;
25 }
26
27 TMatrix::TMatrix(const TMatrix& other) { // конструктор копіювання
28     n = other.n;
29     matrix = new double* [n];
30     for (int i = 0; i < n; i++) {
31         matrix[i] = new double[n];
32         for (int j = 0; j < n; j++) {
33             matrix[i][j] = other.matrix[i][j];
34         }
35     }
36 }
37 TMatrix::TMatrix(int dimention) { // конструктор для випадкового заповнення
38     n = dimention;
39     srand(time(NULL));
40     matrix = new double* [dimention];
41     for (int i = 0; i < dimention; i++) {
42         matrix[i] = new double[dimention];
43         for (int j = 0; j < dimention; j++) {
44             matrix[i][j] = rand() % 10;
45         }
46     }
47 }

```

```

Лаб_4.cpp  Functions.cpp  Class.cpp*  Functions.h  Class.h
Лаб_4  TMatrix  TMatrix(int dimention, do

47  TMatrix::TMatrix(int dimention, double number) { // конструктор для заполнения от нашего числа
48      n = dimention;
49      matrix = new double* [dimention];
50      for (int i = 0; i < dimention; i++) {
51          matrix[i] = new double[dimention];
52          for (int j = 0; j < dimention; j++) {
53              matrix[i][j] = number++;
54          }
55      }
56  }
57  TMatrix::~TMatrix() {
58      for (int i = 0; i < n; i++) {
59          delete[] matrix[i];
60      }
61      delete[] matrix;
62  }
63  void TMatrix::findNormaM() { // поиск нормы M (максимальная из сумм рядов матрицы)
64      double sum, max = 0;
65      for (int i = 0; i < n; i++) {
66          sum = 0;
67          for (int j = 0; j < n; j++) {
68              sum += abs(matrix[i][j]);
69          }
70          if (max < sum) max = sum;
71      }
72      setNormaM(max); // normaM=max;
73  }
74  void TMatrix::findNormal() { // поиск нормы L (максимальная из сумм столбцов матрицы)
75      double sum, max = 0;
76      for (int i = 0; i < n; i++) {
77          sum = 0;
78          for (int j = 0; j < n; j++) {
79              sum += abs(matrix[j][i]);
80          }
81          if (max < sum) max = sum;
82      }
83      setNormal(max);
84  }
85  }
86  // поиск Эвклидовой нормы (корень из сумм квадратов всех элементов)
87  void TMatrix::findNormaE() {
88      double sum = 0;
89      for (int i = 0; i < n; i++) {
90          for (int j = 0; j < n; j++) {
91              sum += pow(matrix[i][j], 2);
92          }
93      }
94      sum = sqrt(sum);
95      setNormaE(sum);
96  }
97  }

```

```

Лаб_4.cpp  Functions.cpp  Class.cpp  Functions.h  Class.h
Лаб_4  TMatrix  findNormaE()

97 void TMatrix::print() { // Вывод
98     for (int i = 0; i < n; i++) {
99         for (int j = 0; j < n; j++) {
100             cout << setw(9) << matrix[i][j];
101         }
102         cout << endl;
103     }
104     cout << "Norm M = " << normaM << endl;
105     cout << "Norm L = " << normaL << endl;
106     cout << "Norm E = " << normaEuclid << endl << endl;
107 }
108
109 TMatrix TMatrix::operator+(const TMatrix& obj) const { // перегрузка оператора плюса
110     TMatrix tmp(n, 1);
111     for (int i = 0; i < n; i++) {
112         for (int j = 0; j < n; j++) {
113             tmp.matrix[i][j] = matrix[i][j] + obj.matrix[i][j];
114         }
115     }
116     return tmp;
117 }
118
119 TMatrix TMatrix::operator*(const TMatrix& obj) const { // перегрузка оператора множення
120     TMatrix tmp(n, 1);
121     double s;
122     for (int i = 0; i < n; i++) {
123         for (int j = 0; j < n; j++) {
124             s = 0;
125             for (int k = 0; k < n; k++) {
126                 s += matrix[i][k] * obj.matrix[k][j];
127             }
128             tmp.matrix[i][j] = s;
129         }
130     }
131     return tmp;
132 }
133
134 void TMatrix::operator=(const TMatrix& obj) { // перегрузка оператора равенства
135     for (int i = 0; i < n; i++) {
136         delete[] matrix[i];
137     } delete[] matrix;
138     n = obj.n;
139     matrix = new double* [n];
140     for (int i = 0; i < n; i++) {
141         matrix[i] = new double[n];
142         for (int j = 0; j < n; j++) {
143             matrix[i][j] = obj.matrix[i][j];
144         }
145     }
146 }

```

```

Лаб_4.cpp    Functions.cpp    Class.cpp
Лаб_4
1      #pragma once
2      #include "Class.h"
3
4
5      void invalidInput(int&);
6      void invalidInput(double&);
7      int input();
8      double inputNum();
9      void findNorm(TMatrix&);
10     double** inputM(int);
    
```

```

Лаб_4.cpp    Functions.cpp    Class.cpp    Functions.h    Class.h
Лаб_4    TMatrix
1      #pragma once
2      #include <iostream>
3      #include <iomanip>
4
5      using namespace std;
6
7      class TMatrix {
8      public:
9          int n;
10         double num;
11         double** matrix;
12         double normaM, normal, normaEuclid;
13     public:
14         double getEl(int i, int j);
15         double getNormaM();
16         void setNormaM(double m);
17         double getNormal();
18         void setNormal(double l);
19         double getNormaE();
20         void setNormaE(double euclid);
21         TMatrix(const TMatrix& other); // копіювання
22         TMatrix(int dimension); // рандом
23         TMatrix(int dimension, double number); // по нашому числу
24         TMatrix(int dimension, double** matr) : n(dimension), matrix(matr) {}
25         ~TMatrix();
26         void findNormaM(); // ряди
27         void findNormaL(); // стовбці
28         void findNormaE(); // квадрати
29         void print();
30         TMatrix operator+(const TMatrix& obj) const;
31         TMatrix operator*(const TMatrix& obj) const;
32         void operator=(const TMatrix& obj);
    };
    
```

2. Результат виконання на C++

```
Enter the dimention of your M1: лошгротл
Your input is wrong, try again: 0
Your input is wrong, try again: 3

Enter the number you want to fill M2 with: juh
Your input is wrong, try again: 2.3

Your randomly generated M1:
    0      8      1
    7      9      0
    3      6      1
Norm M = 16
Norm L = 23
Norm E = 15.5242

Your M2 filled with 2.3:
    2.3     3.3     4.3
    5.3     6.3     7.3
    8.3     9.3    10.3
Norm M = 27.9
Norm L = 21.9
Norm E = 20.4257

M3[0][0] = 0
M3[0][1] = -1.2
M3[0][2] = 5.1
M3[1][0] = 3
M3[1][1] = -6
M3[1][2] = 0.8
M3[2][0] = 7.9
M3[2][1] = 5
M3[2][2] = 0.4

Your own M3:
    0     -1.2     5.1
    3      -6     0.8
    7.9      5     0.4
Norm M = 13.3
Norm L = 12.2
Norm E = 12.6752
```

```
Sum M1 + M2:
    2.3     11.3     5.3
    12.3     15.3     7.3
    11.3     15.3    11.3
Norm M = 37.9
Norm L = 41.9
Norm E = 33.0032

Squared M3:
    204.17   279.97   154.57
    298.97   484.77   259.37
    341.87   534.67   299.27
Norm M = 1175.81
Norm L = 1299.41
Norm E = 1013.87

First (random) M1:
    0      8      1
    7      9      0
    3      6      1
Norm M = 16
Norm L = 23
Norm E = 15.5242

Second (filled) M2:
    2.3     3.3     4.3
    5.3     6.3     7.3
    8.3     9.3    10.3
Norm M = 27.9
Norm L = 21.9
Norm E = 20.4257

C:\Users\Пользователь\OneDrive
```

3. Висновок

Під час виконання четвертої лабораторної роботи, розглянули на практиці роботу з класами з використанням перевантажених операторів. В результаті була створена програма, яка за допомогою різних конструкторів ініціалізує три об'єкти класу «Квадратна матриця», визначає матрицю M3 як суму матриць M1 та M2, а потім підносить її до квадрату.