**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Основы информатики»

Отчет по лабораторной работе №7

# «Обработка и печать числовой матрицы»

Вариант 7

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-13 |  | преподаватель каф. ИУ5 |
| Коновалов Максим |  | Папшев И.С |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2020 г.

## Содержание отчета

Отчет по лабораторной работе состоит из 4-х разделов, отражающих основные этапы разработки программы:

- Постановка задачи;

- Разработка алгоритма;

- Кодирование (соответствующий раздел отчета называется «Текст программы»);

- Тестирование (соответствующий раздел отчета называется «Анализ результатов»).

Раздел «Разработка алгоритма» содержит схему алгоритма функции печати матрицы.

В разделе «Текст программы» приведен листинг программы, включающий необходимые комментарии.

## Постановка задачи;

2.1 Разработать функцию для вывода на экран матрицы double matr[N][M] для распечатки двухмерных массивов. Размерность матрицы может быть произвольной.

Функция должна обеспечивать удобное для пользователя отображение матрицы на экране в (фиксированном (fixed) или )научном (scientific) формате с требуемой точностью ***n*** знаков после запятой (устанавливается setprecision(n)). Целые числа также выводятся в научном формате. Матрица,(тип формата,) число знаков после запятой должны передаваться в функцию в виде параметров (число знаков в целой части для научного формата всегда равно 1).

*Указание:* В зависимости от размера матрицы и требуемого числа печатаемых знаков после запятой, функция должна вычислить ширину поля вывода одного элемента матрицы ***w*** (при печати используется в операторе setw(w) ) и определить число столбцов матрицы (ncol), которое поместится в одной строке на экране.

Строка на экране вмещает 80 символов и все столбцы матрицы могут не поместиться в одной строке. В этом случае печать матрицы следует выполнять в несколько рядов по ncol столбцов в каждом (кроме последнего ряда). В каждом ряду печатайте все строки столбцов этого ряда. Над каждым столбцом напечатайте его номер.

2.2 Используйте разработанную в п. 2.1 функцию для печати квадратной матрицы А размером N\*N (где N вводится с клавиатуры).

- все элементы главной диагонали матрицы А равны 1;

- элементы, лежащие выше главной диагонали, вычисляются по формуле

**A i,j = xi / (j!)i ,**

а элементы, лежащие ниже главной диагонали, по формуле

**A i,j = (-x)i / (j!)i**, где **i,j =1,2,…,N**; **x=1**.

Для вычисления значений элементов матрицы используйте ***рекуррентные соотношения*** (без факториалов и степеней).

Создание и заполнение матрицы реализуйте в виде ***отдельных функций***.

(2.3 Не изменяя кода функции печати матрицы, распечатать матрицу в «научном» формате и в формате с фиксированной точкой с требуемой точностью.)

2.3 Распечатать с помощью разработанной функции целочисленную статическую (заданную с помощью оператора описания) матрицу размером **B[10][10].** Для передачи матрицы в функцию печати создайте вспомогательный массив указателей на строки матрицы **B** . Значения элементов матрицы **В** определяются соотношением: **B[i][j]=i\*10+j**.

2.5 Создайте шаблон для функции печати числовых матриц и выполните п.2.2 и 2.3 с использованием шаблона;

2.5 Вставьте в программу и объясните результаты выполнения следующих операторов

для матрицы В[10][10]:

cout<<B<<" "<<B[0]<<" "<<B[2]<<endl;

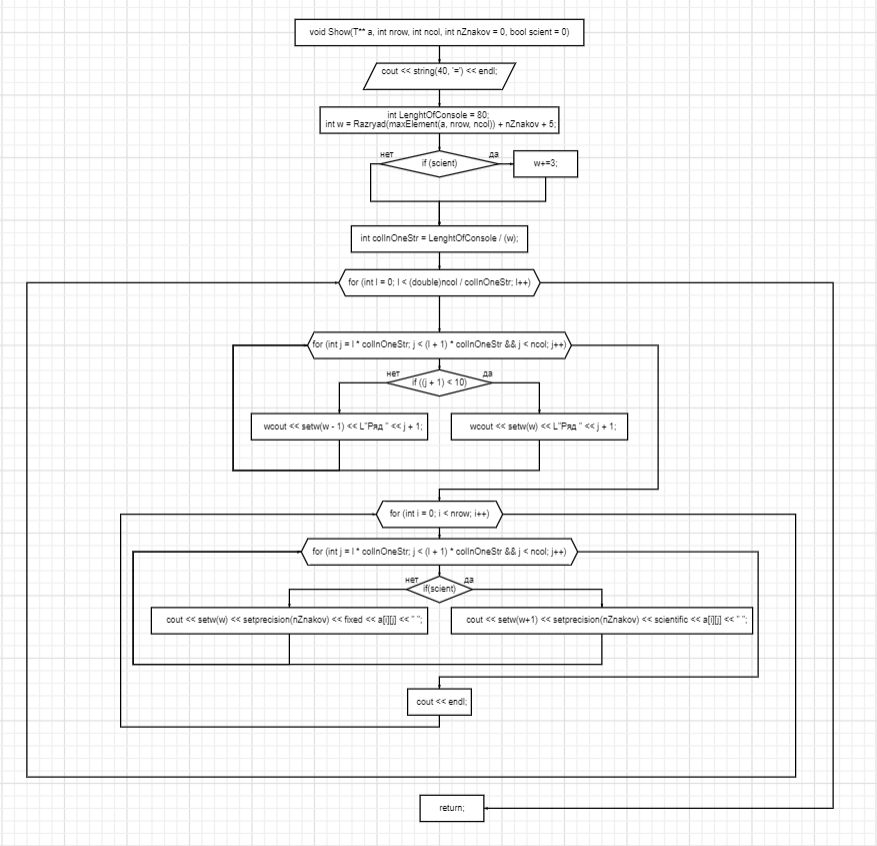
cout<<B[0][0]<<" "<<\*\*B <<" "<<\*B[0]<<endl;

cout<<\*(\*(B+1))<<" "<<\*B[1]<<endl;

cout<<\*(B[0]+1)<<" " <<\*(\*B+1)<<endl;

cout<<B[0][20]<<" "<<\*(B[0]+20)<<" "<<\*B[2]<<endl;

## Разработка алгоритма;



## Текст программы;

### Main.cpp

#include <iostream>

#include <iomanip>

#include "PrintMatrix.h"

using namespace std;

int sign(int a)

{

if (a == 0) return 0;

return (a > 0) ? 1 : -1;

}

const int B[10][10] = {

0, 1, 2, 3, 4, 5, 6, 7, 8, 9,

10, 11, 12, 13, 14, 15, 16, 17, 18, 19,

20, 21, 22, 23, 24, 25, 26, 27, 28, 29,

30, 31, 32, 33, 34, 35, 36, 37, 38, 39,

40, 41, 42, 43, 44, 45, 46, 47, 48, 49,

50, 51, 52, 53, 54, 55, 56, 57, 58, 59,

60, 61, 62, 63, 64, 65, 66, 67, 68, 69,

70, 71, 72, 73, 74, 75, 76, 77, 78, 79,

80, 81, 82, 83, 84, 85, 86, 87, 88, 89,

90, 91, 92, 93, 94, 95, 96, 97, 98, 99

};

const int\* pB[] = { B[0], B[1], B[2], B[3], B[4], B[5], B[6], B[7], B[8], B[9] }; //массив указателей на массивы

const int x = 1.0; // из методички

int menu()

{

cout << "================ВЫВОД МАТРИЦ===================\n";

cout << "\t1 - Вывод матрицы А\n";

cout << "\t2 – Вывод матрицы B\n";

cout << "\t3 - Выход\n";

int choice;

cout << "Выберите действие\n";

cin >> choice;

while (cin.fail()) { //исключение зацикливания в случае ввода недопустимых символов

cout << "Ошибка ввода. Повторите ввод\n";

cin.clear(); //сброс в потоке флага ошибки

cin.ignore(10, '\n'); //извлечение из буфера потока введенных символов

cin >> choice; //повторный ввод

}

return choice;

}

int main()

{

double\*\* A; // Матрица А

setlocale(LC\_ALL, "Russian");

while (true) {

switch (menu()) {

case 1:

int N;

cout << "Введите N:";

cin >> N;

bool scient;

cout << "В каком виде вывести матрицу?\n 0 - Fixed \n 1 - Scientific\n";

cin >> scient;

cout << "Введите кол-во знаков после запятой : ";

int nZnakov;

cin >> nZnakov;

A = new double\* [N];

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

A[i] = new double[N];

A[0][0] = -x;

A[0][1] = x / 2.0;

for (int j = 2; j < N; j++)

A[0][j] = A[0][j - 1] / (j + 1);

for (int i = 1; i < N; i++) {

A[i][0] = A[i - 1][0] \* -x;

for (int j = 1; j < N; j++) {

if (j >= i) {

A[i][j] = A[i - 1][j] \* A[0][j];

}

else {

A[i][j] = abs(A[i - 1][j] \* A[0][j]) \* sign(A[i][0]);

}

}

}

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

A[i][i] = 1;

Show(A, N, N, nZnakov, scient);

break;

case 2:

Show(pB, 10, 10);

break;

case 3:

return 0; //завершение работы

default:

cout << "Недопустимое действие.Повторите выбор.\n";

}

}

}

### PrintMatrix.h

#pragma once

#include <iomanip>

#include <iostream>

#include <string>

using namespace std;

int Razryad(double a)

{

int k = (int)a;

int i = 0;

while (k > 0)

{

k /= 10;

i++;

}

return i;

}

int Razryad(int a)

{

int k = (int)a;

int i = 0;

while (k > 0)

{

k /= 10;

i++;

}

return i;

}

int Razryad(long double a)

{

int k = (int)a;

int i = 0;

while (k > 0)

{

k /= 10;

i++;

}

return i;

}

template <typename T>

double maxElement(T\*\* a, int nrow, int ncol) {

double max = NULL;

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

{

for (int j = 0; j < ncol; j++)

{

a[i][j] > max ? max = a[i][j] : false;

}

}

return max;

}

template <typename T>

void Show(T\*\* a, int nrow, int ncol, int nZnakov = 0, bool scient = 0)

{

cout << string(40, '=') << endl;

int LenghtOfConsole = 80;

int w = Razryad(maxElement(a, nrow, ncol)) + nZnakov + 5;

if (scient) w += 3;

int colInOneStr = LenghtOfConsole / (w);

for (int l = 0; l < (double)ncol / colInOneStr; l++)

{ //Печатаем ряд

for (int j = l \* colInOneStr; j < (l + 1) \* colInOneStr && j < ncol; j++) {

if ((j + 1) < 10)

wcout << setw(w) << L"Ряд " << j + 1;

else

{

wcout << setw(w - 1) << L"Ряд " << j + 1;

}

}

cout << endl;

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

for (int j = l \* colInOneStr; j < (l + 1) \* colInOneStr && j < ncol; j++) {

if (scient)

cout << setw(w+1) << setprecision(nZnakov) << scientific << a[i][j] << " ";

else

cout << setw(w) << setprecision(nZnakov) << fixed << a[i][j] << " ";

}

cout << endl;

}

}

}

## Анализ результатов.

