Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра “Електронних обчислювальних машин”



**Звіт з лабораторної роботи №3**

на тему:

**“ ЗАСОБИ РОБОТИ З ДИНАМІЧНОЮ ПАМ’ЯТТЮ.**

**ДИНАМІЧНІ МАСИВИ”**

**Виконала:**

ст. гр. КІ-15

Аркавенко Л.С.

**Перевірив:**

асист. каф. ЕОМ

Козак Н. Б.

Львів – 2020

**Мета роботи:**

познайомитися із динамічними масивами.

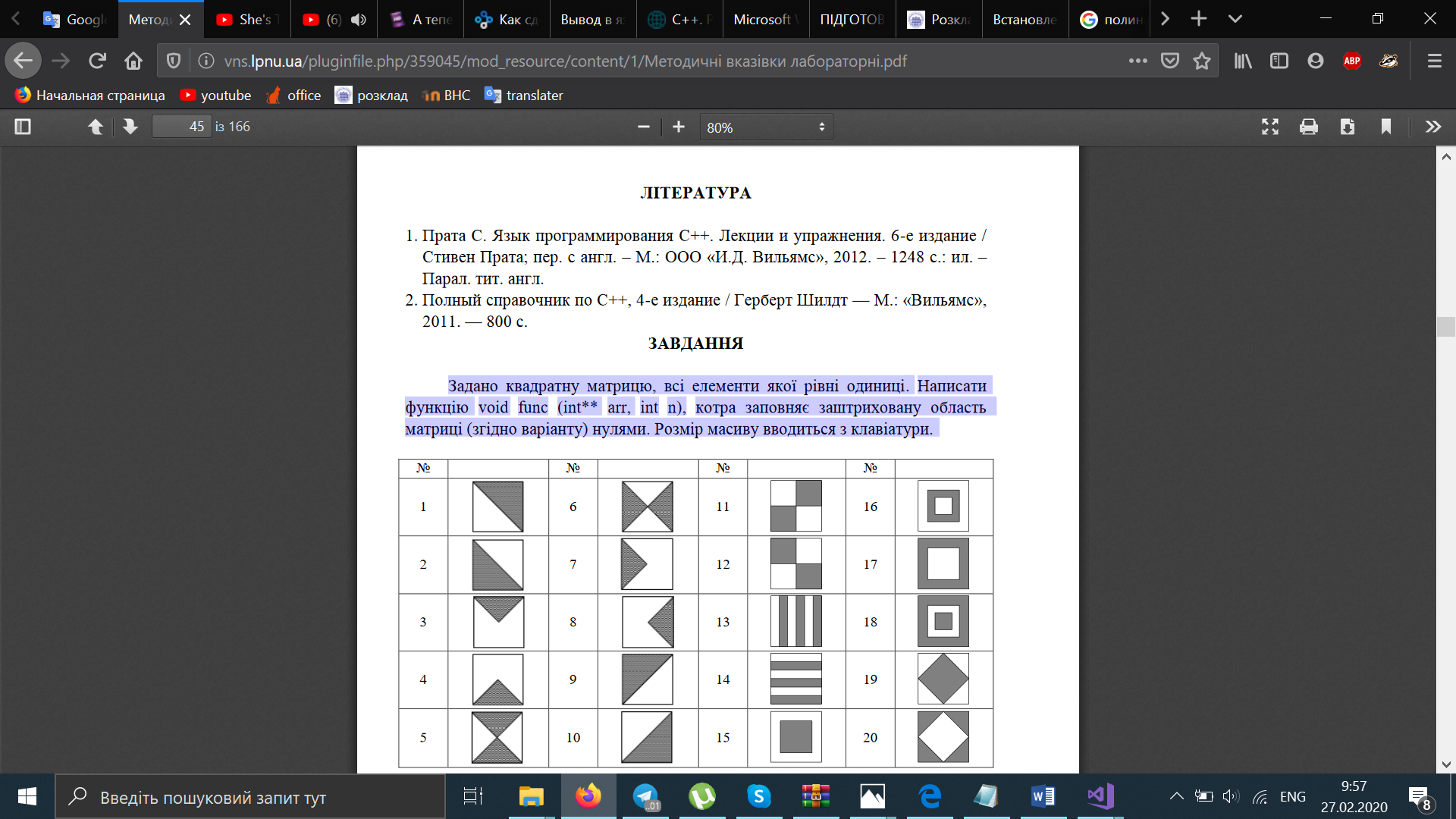
**Теоретична частина:**

Динамічне виділення пам’ятіВ С++ об‘єкти можна розміщати статично –під час компіляції, абодинамічно –під час виконання програми, шляхом виклику функцій зі стандартної бібліотеки. Основна відмінність у використанні даних методів –в їхній ефективності та гнучкості. Статичне розміщення більш ефективне, так як виділення пам‘яті відбувається до виконання програми, проте воно менш гнучке, тому що необхідно наперед знати тип і розмір об‘єкту. Задачі, в яких необхідно зберігати та обробляти наперед не відому кількість елементів, зазвичай потребують динамічного виділення пам‘яті.Динамічне виділення пам‘яті у мові С++ здійснюється за допомогою оператора new.Оператор newздійснює пошук неперервної області пам‘яті в області пам‘яті, що зветься некерована куча. Некерована куча –цестрктура даних за допомогою якої реалізована пам‘ять, що може бути виділена динамічнов ході виконання програми, а такожцеобласть пам‘яті, зарезервована під цю структуру.З іншої сторони куча -це довгий відрізок адрес пам'яті, поділений на блоки різних розмірів, що йдуть підряд.Пам‘ять у кучі поділяється на заняту і вільну.Перед початком роботипрограми вся пам‘ять у кучі позначається як вільна.При виклику оператора динамічного виділення пам‘ятіу кучівідбувається пошук неперервного сегменту вільної пам‘яті заданого розміру.Час такого пошукує значним ізаймає більшу частину часу, що необхідна для виконання операції динамічного виділення пам‘яті.Якщо такий сегмент було знайдено в кучі, товін помічається як занятий і програмі повертається адреса його початку, інакше –програмі повертається ознакавідсутності такого сегменту в пам‘яті, найчастіше NULL. Якщо в ході виконання програми значення адреси початку цього сегменту втрачається, то заняту пам‘ять звільнити буде неможливо. Якщо ця ситуація проявляється неодноразово, то це може призвести до вичерпання вільної пам‘яті в системі. Коли динамічно виділена область пам‘яті стає непотрібною,

38то її потрібно звільнити за допомогою оператора звільнення динамічно виділеної пам‘яті. При його виклику область пам‘яті на яку вказує вказівник і якабулапопередньо динамічно виділена з кучі позначається як вільна і її можна буде в подальшому використовувати заново.

**Індивідуальне завдання:**

Задано квадратну матрицю, всі елементи якої рівні одиниці. Написати функцію voidfunc(int\*\* arr, intn),котра заповняє заштриховану область матриці (згідно варіанту) нулями. Розмір масиву вводиться з клавіатури.



**Хід роботи:**

**Код програми:**

#include <iostream>

#include <iomanip>

#include <conio.h>

using namespace std;

void fill\_array\_one(int\*\*, size\_t);

void fill\_array\_scheme(int\*\*, size\_t);

void print\_array(int\*\*, size\_t);

int main() {

size\_t dimm;

int \*\*my\_arr;

cout << "Enter the size of matrix:" << endl;

cin >> dimm;

my\_arr = new int \*[dimm];

for (size\_t i = 0; i < dimm; i++)

my\_arr[i] = new int[dimm];

cout << "Your matrix is filled with 1: " << endl;

fill\_array\_one(my\_arr, dimm);

print\_array(my\_arr, dimm);

cout << "Your matrix is filled according to the scheme:" << endl;

fill\_array\_scheme(my\_arr, dimm);

print\_array(my\_arr, dimm);

for (size\_t i = 0; i < dimm; i++)

delete[] my\_arr[i];

delete[] my\_arr;

\_getch();

return 0;

}

void fill\_array\_one(int\*\* arr, size\_t n) {

for (size\_t i = 0; i < n; ++i)

for (size\_t j = 0; j < n; ++j)

arr[i][j] = 1;

}

void fill\_array\_scheme(int\*\* arr, size\_t n) {

for (size\_t i = 0; i < n; ++i)

for (size\_t j = 0; j < n; ++j)

if (i >= j)

arr[i][j] = 0;

}

void print\_array(int\*\* array, size\_t n) {

for (size\_t i = 0; i < n; ++i) {

for (size\_t j = 0; j < n; ++j)

cout << setw(2) << array[i][j];

cout << endl;

\_getch();

}

}

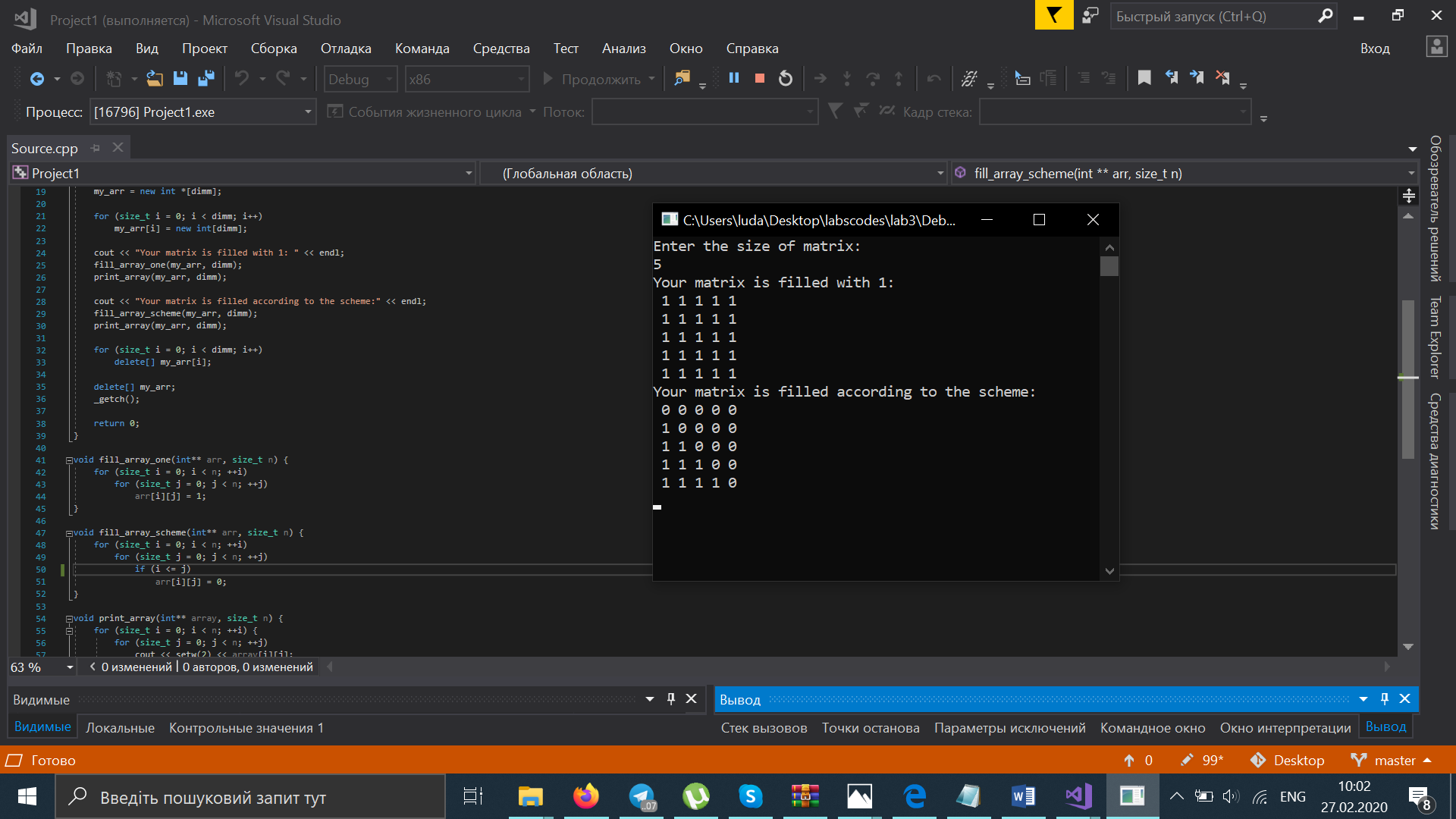


Рис 1. Програма та консольний вивід до завдання.

**Висновок:**

Я познайомилась iз довiдковою системою iнтегрованого середовища Visual C++2017, iз засобами роботи з динамічною пам’яттю та динамічними масивами.