Національний технічний університет України

«Київський політехнічний інститут»

Кафедра автоматизованих систем обробки інформації та управління

Звіт про виконання лабораторної роботи №5 з дисципліни

«Алгоритми та структури даних»

Перевірено: Ковалюк Т.В. Виконав ст. 1 курсу ФОІТ

Проскура С.Л. гр. ІС-52

Дорошенко А.В.

Київ 2016

## Лабораторна робота 5

# **Динамічні структури. Черги**

Варіант № 9

*Мета:*

1. Вивчити особливості використання  динамічних структур типу черг.
2. Навчитися застосовувати черги в практичних задачах

### Завдання

|  |  |
| --- | --- |
| Створити чергу з елементів, значення яких зчитані з текстового файлу. Здійснити перевірку черги на пустоту, копіювання черги, визначення кількості її елементів. Вивести вхідну та вихідну черги та відповідні повідомлення. |  |

### Блок-схема алгоритму



Рис.1 (Блок-схема алгоритмудодавання елемента до черги)



Рис.2 (Блок-схема головного алгоритму)



Рис.3 (Блок-схема копіювання черги та виведення результатів)



Рис.4 (Блок-схема алгоритму видалення елемента з черги і збереження його значення)

### Код програми

//Laboratory work

//written by student of the first curse

//of the group IC-52

//Anton Doroshenko

//2016.04.10

//==========================================================================

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include<iostream>

#include<conio.h>

#include<iomanip>

using namespace std;

//черга

struct Item

{

char val;//значення

Item\* next;//покажчик на наступний елемент

};

Item\* head = NULL;//голова 1 черги

Item\* cur;//поточний елемент

Item\* last = NULL;//кінець 1 черги

Item\* head2 = NULL;//голова 2 черги

Item\* last2 = NULL;//кінець 2 черги

FILE\* f;

char sym;

char\* ptrch = &sym;

int N = 0;//кількість елементів черги

//====== Функція додавання елементу до 1 черги ========

void add()

{

cur = new Item;

cur->val = sym;

cur->next = NULL;

if (last == NULL)

{

head = cur;

}

else

{

last->next = cur;

}

last = cur;

}

//====== Функція додавання елементу до 2 черги ========

void add2()

{

cur = new Item;

cur->val = sym;

cur->next = NULL;

if (last2 == NULL)

{

head2 = cur;

}

else

{

last2->next = cur;

}

last2 = cur;

}

//====== Функція видалення елементу з 1 черги ========

void pop(char\* value)

{

cur = head;

\*value = head->val;

head = cur->next;

if (head == NULL)

{

last = NULL;

}

delete cur;

}

//====== Функція видалення елементу з 2 черги ========

void pop2(char\* value)

{

cur = head2;

\*value = head2->val;

head2 = cur->next;

if (head2 == NULL)

{

last2 = NULL;

}

delete cur;

}

//====== Функція копіювання черги та виведення результатів ========

void copyAndRes()

{

cout << "First queue: ";

while (head != NULL)

{

pop(ptrch);

add2();

cout << sym << " ";

N++;

}

cout << endl;

cout << "Second queue: ";

while (head2 != NULL)

{

pop2(ptrch);

cout << sym << " ";

}

cout << endl;

cout << "The number of elements is " << N << endl;

}

//=========== Головна функція ==============

int main()

{

cout << "laboratory work number 5 made by Anton Doroshenko, IS-52" << endl;

cout << endl;

f = fopen("f.txt", "r");

while ((sym = fgetc(f)) != EOF)

{

add();

}

fclose(f);

if (head == NULL)

{

cout << "Queue is empty" << endl;

}

else

{

cout << "Queue isn`t empty" << endl;

copyAndRes();

}

system("pause");

}

1. **Screen Shot результатів**

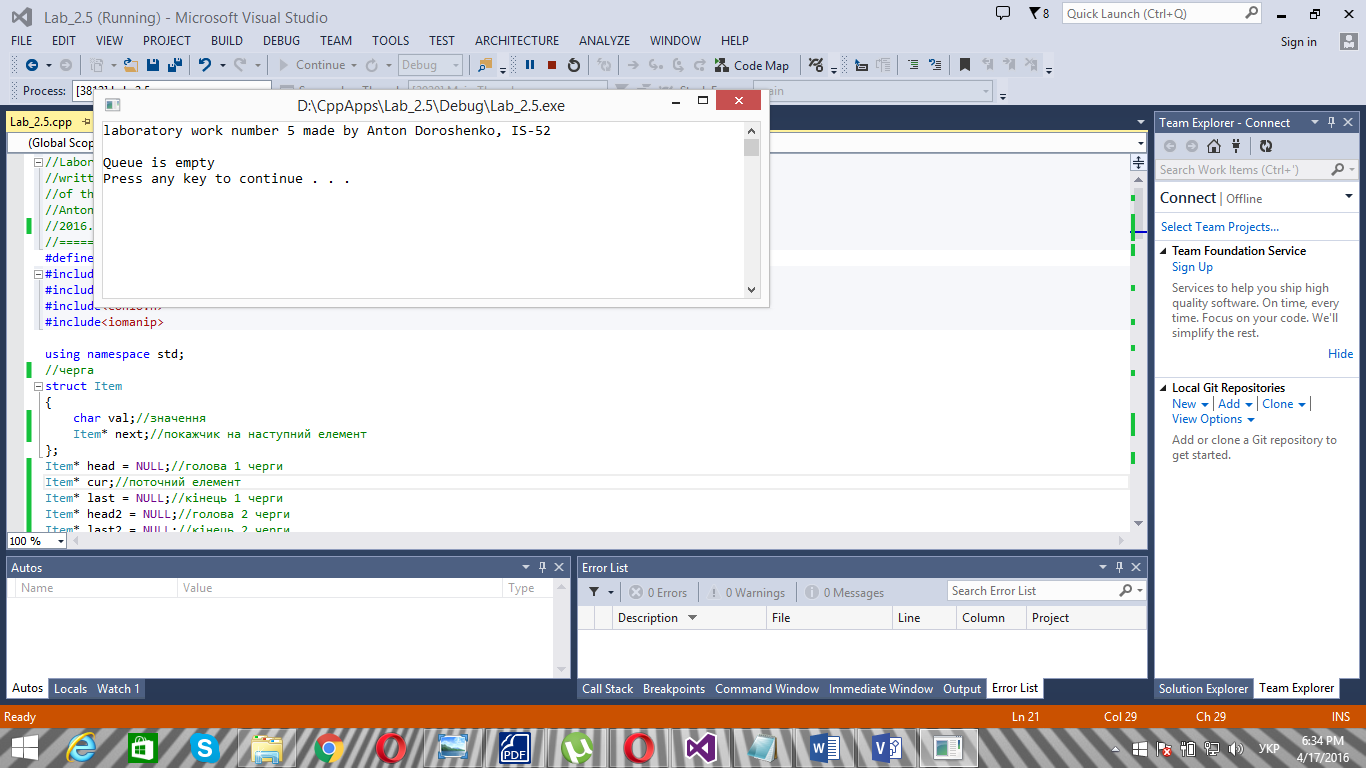


Рис.5 (Screen Shot результатів 1)

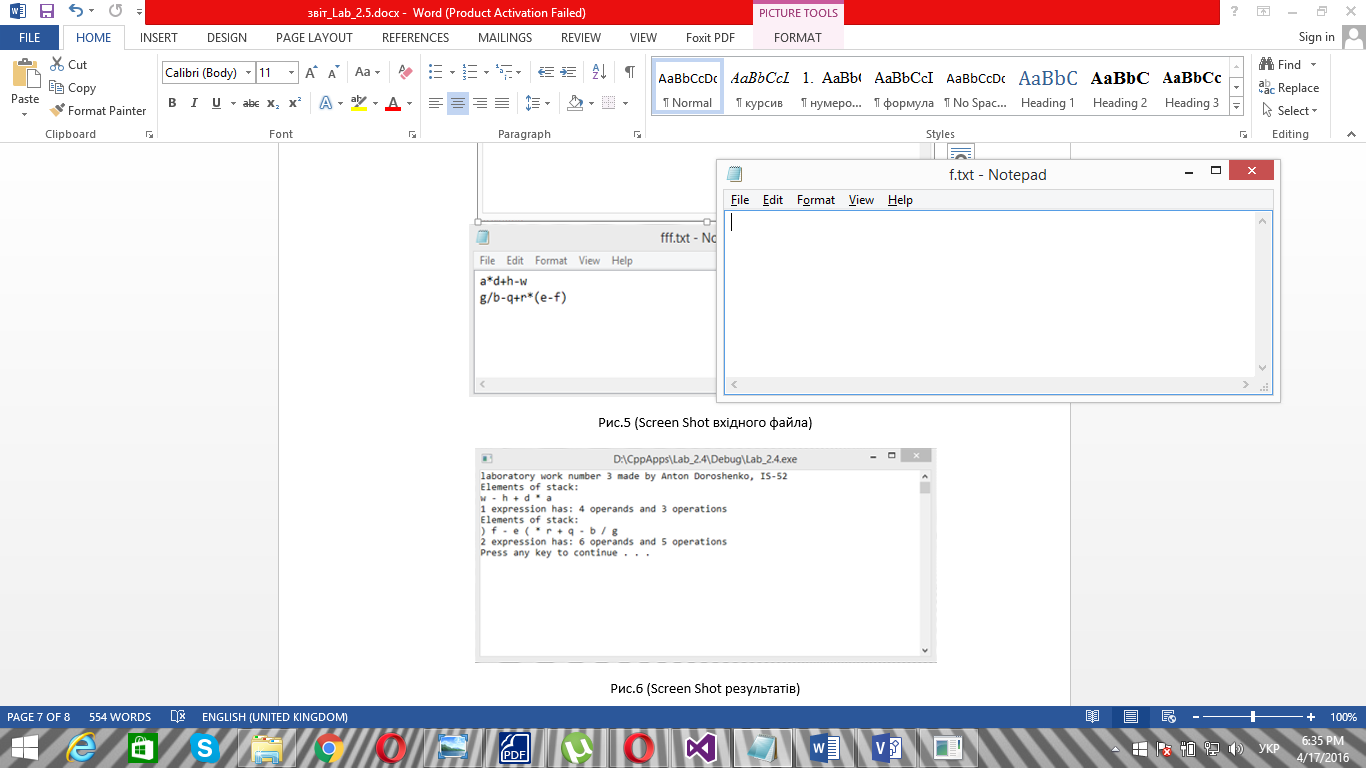


Рис.6 (Screen Shot вхідного файла 1)

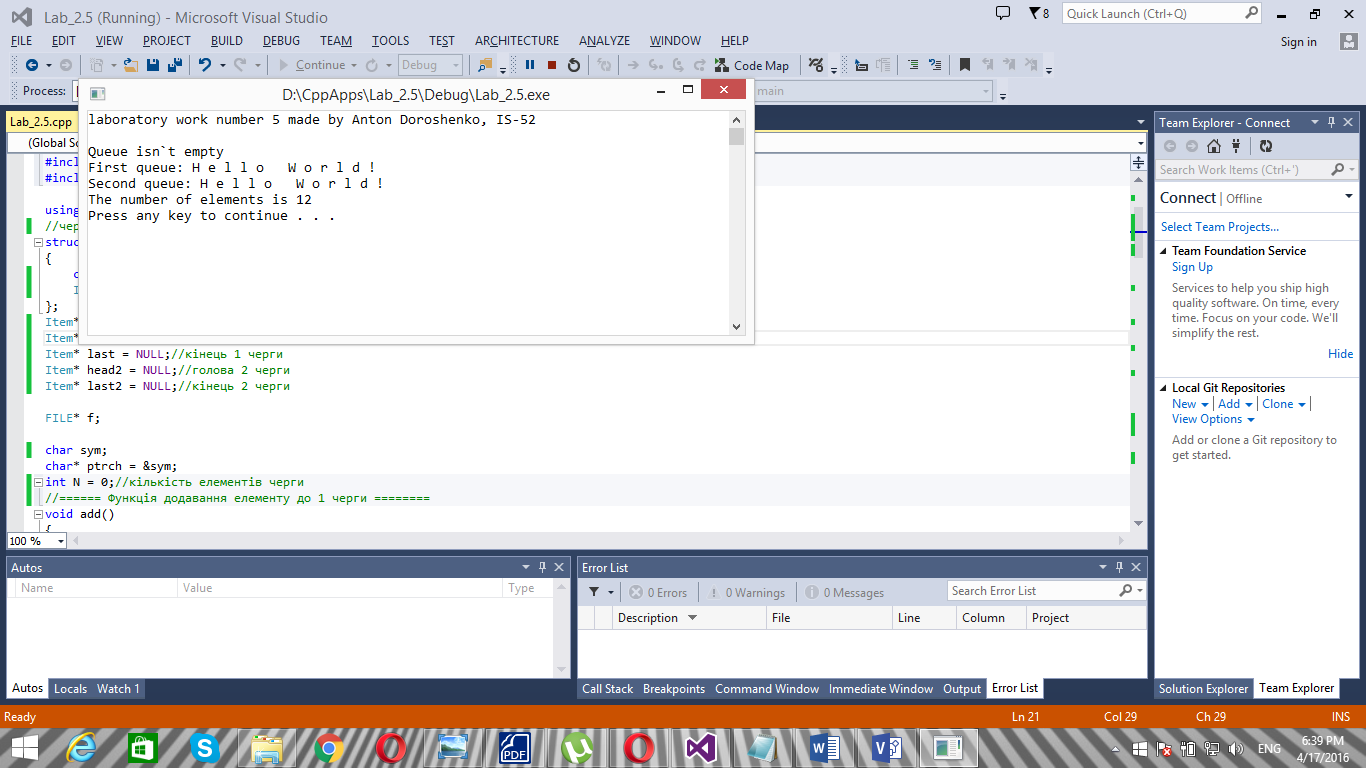


Рис.7 (Screen Shot результатів 2)

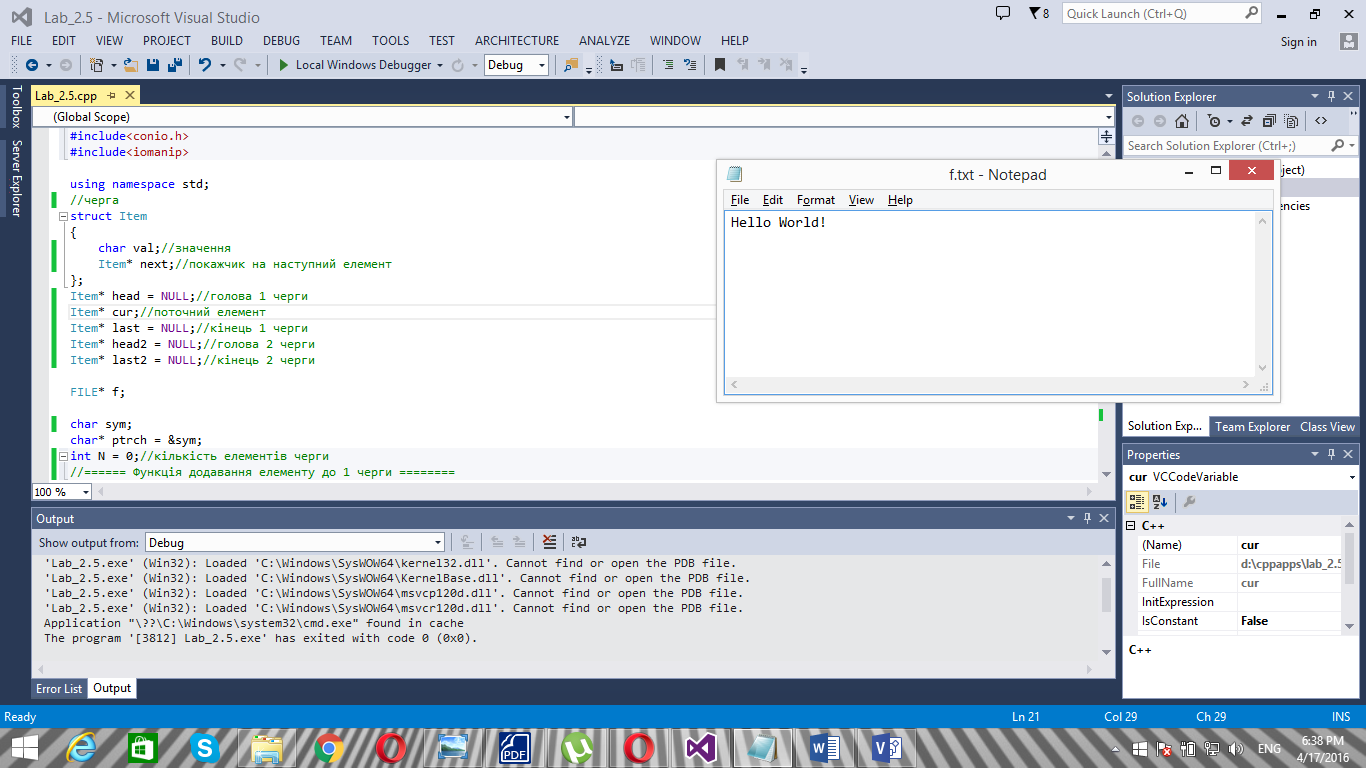


Рис.8(Screen Shot вхідного файла 2)

1. **Аналіз результатів**

Як ми бачимо зі ScreenShot’ів

Програма зчитує дані з файла, записує їх у чергу. Якщо черга не порожня програма робить копію черги, потім виводить на консоль першу та другу черги, а також кількість елементів черги.

1. **Висновок**

Програма працює правильно, про що свідчить аналіз результатів та ScreenShot’и. Програма коректно виводить результати. Програма застосовує динамічну структуру даних черга.