Національний технічний університет України

«Київський політехнічний інститут»

Кафедра автоматизованих систем обробки інформації та управління

Звіт про виконання лабораторної роботи №9 з дисципліни

«Алгоритмизація та програмування»

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

Проскура С.Л. гр. ІС-52

Дорошенко А.В.

Київ 2015

## Лабораторна робота 9

# **Обробка рядків**

Варіант № 9

*Мета:*

1. Ознайомитися з особливостями рядків;
2. Опанувати технологію застосування функцій обробки рядків;
3. Навчитися розробляти алгоритми та програми із застосуванням рядків.

### Завдання

|  |  |
| --- | --- |
| Заданий рядок s1. Після видалення однієї літери з s1 утворюється рядок s2. Після видалення з s1 іншої літери утворюється рядок s3. Потрібно за рядками s2 і s3 відновити рядок s1. |  |

### Блок-схема алгоритму



Рис.1 (Головний алгоритм)



Рис.2 (Алгоритм знаходження індексу елемента, що видаляється)



Рис.3 (Алгоритм видалення елемента)



Рис.4 (Алгоритм знаходження індекса перших неоднакових елементів)



Рис.5 (Алгоритм зсуву масиву вправо)



Рис.6 (Алгоритм відновлення рядка)

### Код програми

//Laboratory work

//written by student of the first curse

//of the group IC-52

//Anton Doroshenko

//2015.12.06

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

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <stdio.h>

using namespace std;

bool flag = false;

char str[15]; //рядок, що вводиться

char str1[15]; //рядок з якого вилучається деякий елемент

char str2[15]; //рядок з якого вилучається інший елемент

char symbol; //символ, що видаляється

char\* p; //покажчик на шуканий символ

int ind = 0; //індекс символу в рядку

//====== Функція пошуку індекса елемента, що видаляється ======

void SearchSymIndex(char\* string)

{

do

{

cout << "Enter symbol, which you want to delete" << endl;

cin >> symbol;

p = strchr(string, symbol);

ind = p - string;

if (p == 0)

{

cout << "The symbol is not found\nPlease try again" << endl;

}

} while (p == 0);

}

//====== Функція видалення заданого символу ======

void DeleteSymbol(int index, char\* string)

{

int n = strlen(string) - 1; //індекс останньго елемента

for (int i = index; i < n; i++)

{

string[i] = string[i + 1];

}

string[n] = '\0';

}

//====== Функція пошуку індексу перших різних елементів ======

void SearchDifSymInd(char\* string1, char\* string2)

{

ind = 0;

int n = strlen(string1) - 1; //індекс останнього елемента

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

{

if (string1[i] != string2[i])

{

flag = true;

break;

}

ind++;

}

}

//====== Функція зсуву частини рядка вправо ======

void RightReplace(int index, char\* string)

{

int n = strlen(string) - 1;

for (int i = n; i >= index; i--)

{

string[i + 1] = string[i];

}

}

//====== Функція відновлення рядка ======

void Recovery(int index)

{

char temp;

int cmp = 0;

if (index == strlen(str1) - 1)//якщо неоднакові

{ //елементи останні

//видалені елементи суміжні тому буде два варіанти відновлення

RightReplace(index, str2);

str2[index] = str1[index];//вставку елементу з першого рядка в другий

cout << str2 << endl; //виведення 1 варіанта

//перестановка двох елементів

temp = str2[index + 1];

str2[index + 1] = str2[index];

str2[index] = temp;

cout << "or\n" << str2 << endl;//виведення 2 варіанта

}

else if (index != strlen(str1) - 1)

{

cmp = strcmp(str1 + index + 1, str2 + index + 1);

if (cmp == 0)

{

RightReplace(index, str2);

str2[index] = str1[index];

cout << str2 << endl;

temp = str2[index + 1];

str2[index + 1] = str2[index];

str2[index] = temp;

cout << "or\n" << str2 << endl;//виведення 2 варіанта

}

else

{

if (str2[index + 1] == str1[index])

{

RightReplace(index, str1);

str1[index] = str2[index];

cout << str1 << endl;

}

else

{

RightReplace(index, str2);

str2[index] = str1[index];

cout << str2 << endl;

}

}

}

}

//====== Головна функція ======

int main()

{

cout << "laboratory work number 9 made by Anton Doroshenko, IS-52" << endl;

cout << "Enter string" << endl;

gets(str); //введення рядка

strcpy(str1, str); //копіювання рядка

strcpy(str2, str); //копіювання рядка

SearchSymIndex(str1);

DeleteSymbol(ind, str1);

SearchSymIndex(str2);

DeleteSymbol(ind, str2);

cout << "Strings after deleting:" << endl;

cout << str1 << endl;

cout << str2 << endl;

SearchDifSymInd(str1, str2);

if (flag == true)

{

cout << "Recovering string:" << endl;

Recovery(ind);

}

else

{

cout << "The same elements are deleting, string can't be recovered" << endl;

}

}

1. **Screen Shot результатів**

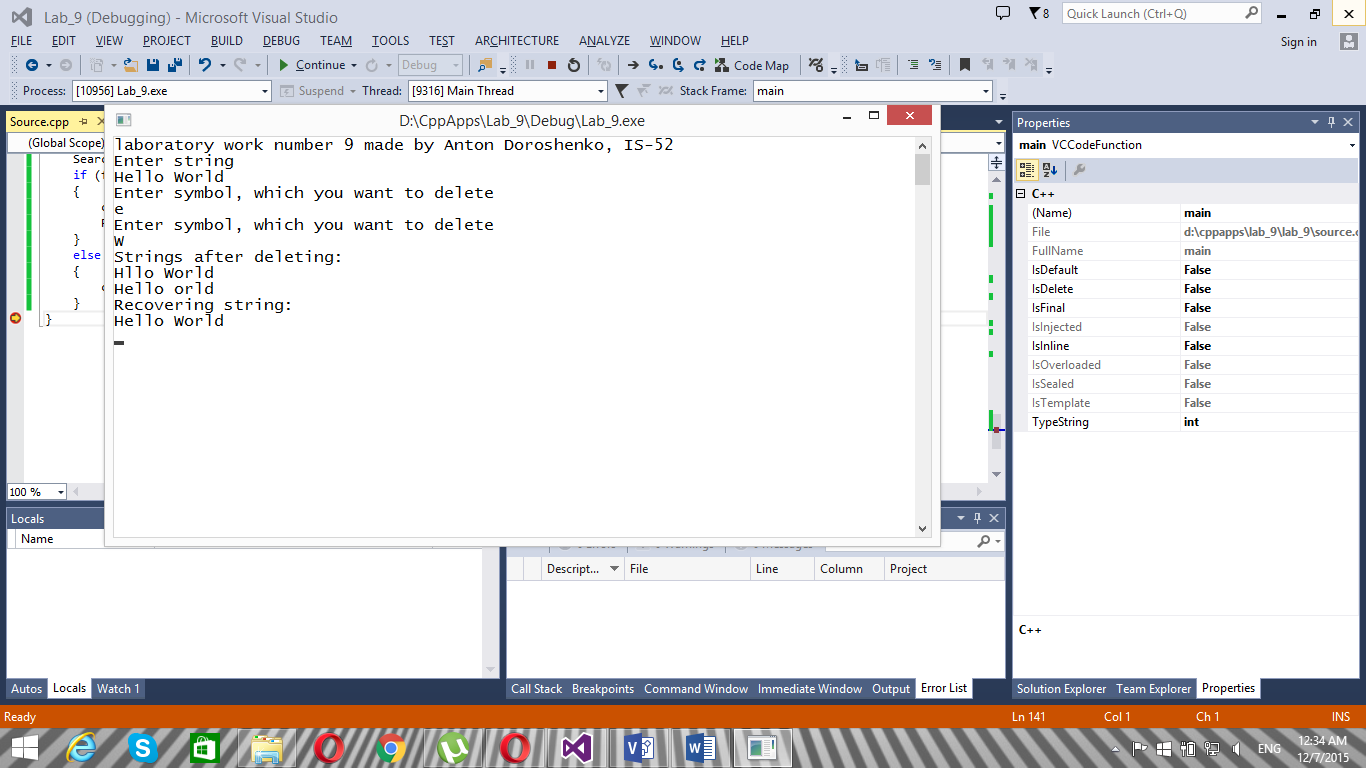


Рис.7 (Screen Shot результатів 1)

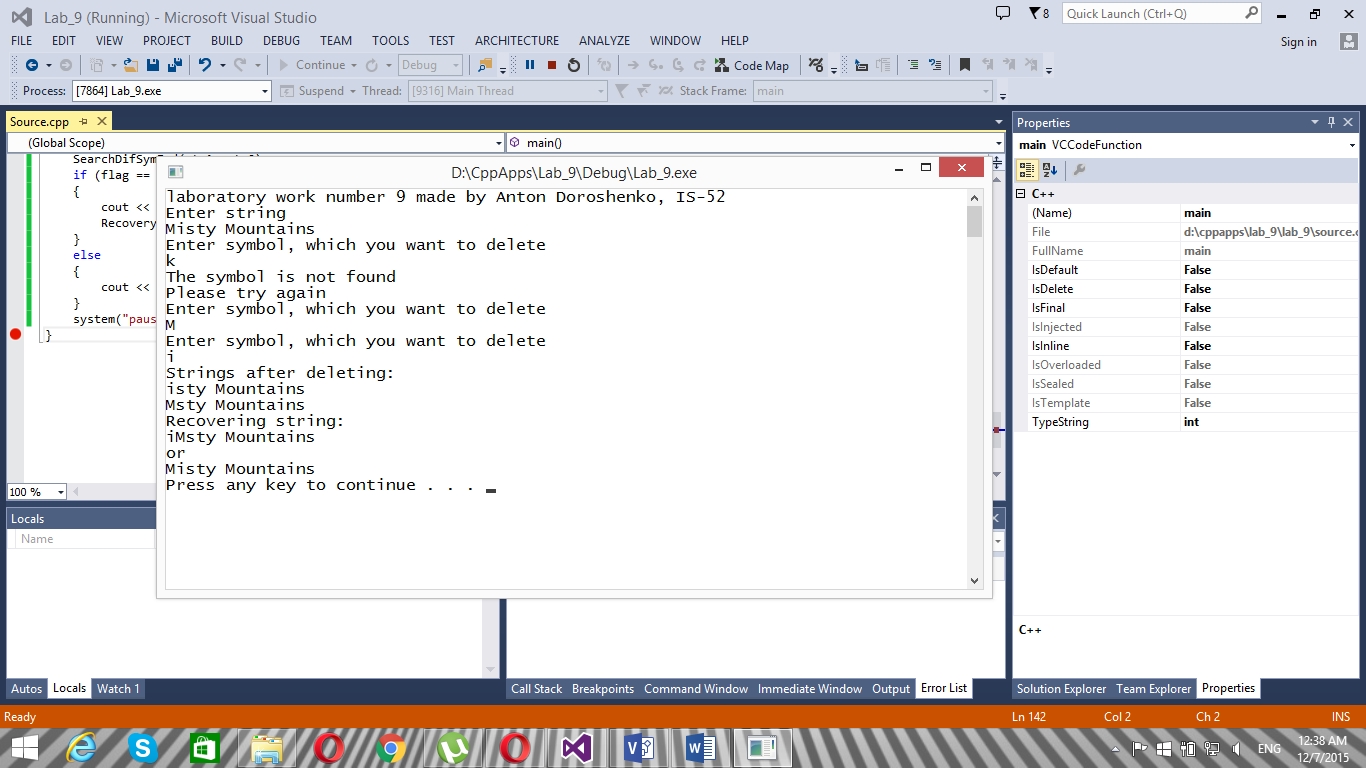


Рис.8 (Screen Shot результатів 2)

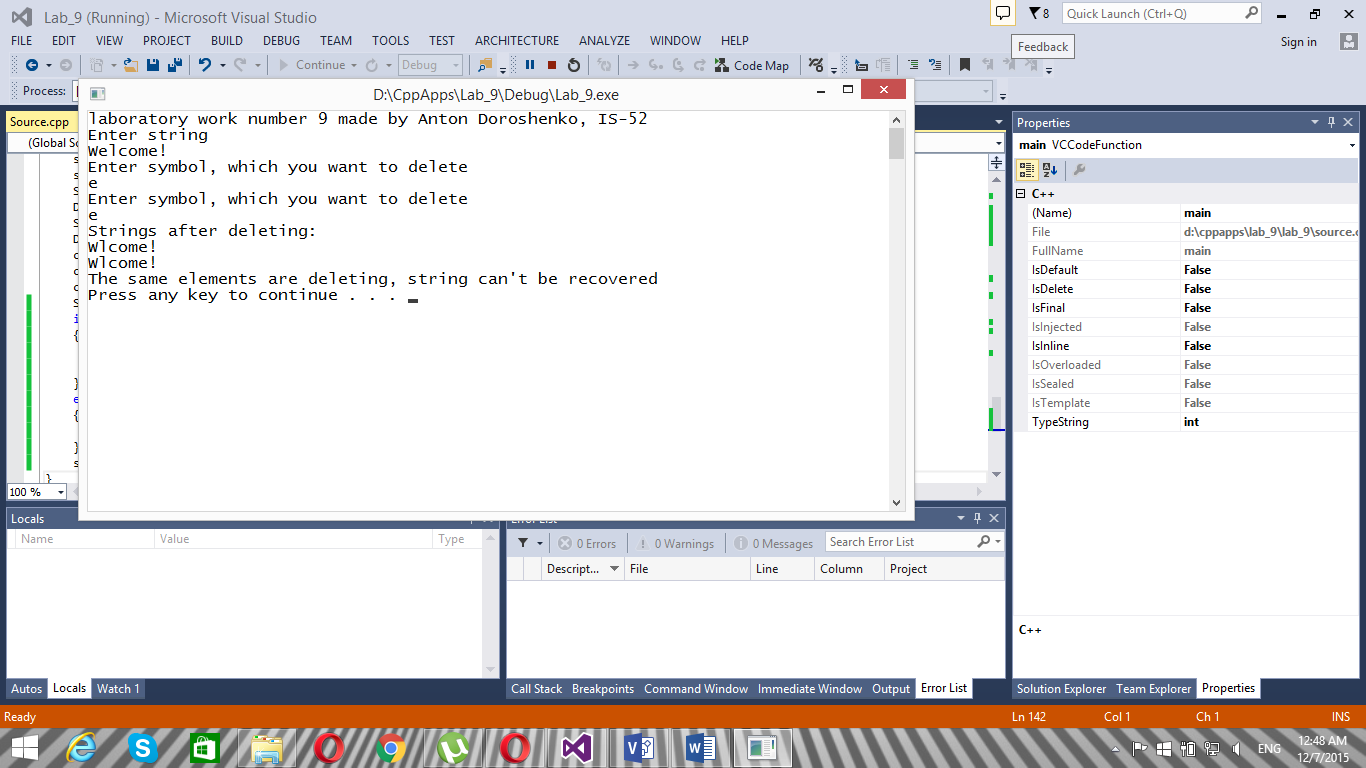


Рис.9 (Screen Shot результатів 3)

1. **Аналіз результатів**

Як ми бачимо зі ScreenShot’ів, ми ввели 3 варіанти даних.

Якщо видаляти елементи, що не стоять поруч у рядку, то виводиться один правильний варіант відновленого рядка. Якщо видаляти елементи, що стоять поруч, то виводяться два варіанти один з яких правильний. Якщо вводити символ якого немає в рядку, то виведеться повідомлення про те що символ не знайдено і програма запропонує спробувати ще раз. Якщо видалені однакові символи виводиться повідомлення про не можливість відновити рядок.

1. **Висновок**

Програма працює правильно, про що свідчить аналіз результатів та ScreenShot’и. Програма коректно виводить результати. Програма захищена від різних виключних ситуацій.