**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

Інститут **ІКНІ**

Кафдера **ПЗ**



**ЗВІТ**

**З дисципліни:** *“Об’єктно-орієнтоване програмування”*

**На тему:** *“Робота з динамічною пам’яттю”*

**Лектор:**

Доцент кафедри ПЗ

Коротєєва Т.О.

**Виконав:**

Ст. гр. ПЗ-11

Мартинюк Н.В.

**Прийняла:**

Доцент кафедри ПЗ

Коротєєва Т.О.

«\_\_» \_\_\_\_\_\_\_\_\_ 2020 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів-2020

**Тема:** Робота з динамічною пам’яттю.

**Мета:** Навчитися виділяти місце під об’єкти динамічно. Навчитися створювати та використовувати конструктор копіювання, перевантажувати оператор присвоєння. Ознайомитися з принципами створення та функціонування деструкторів.

**Завдання для лабораторної роботи:**

1. Переглянути лістинг коду в прикладі. Пояснити вивід програми.
2. Створити клас відповідно до завдання.
3. Розробити для класу конструктор за замовчуванням та декілька звичайних конструкторів. Реалізувати функції-члени відповідно до завдання.
4. Створити конструктор копіювання.
5. Перевантажити операцію присвоєння.
6. Створити деструктор для вивільнення динамічно виділеної пам’яті.
7. Продемонстувати розроблені можливості класу завдяки створеному віконному застосуванню.
8. Оформити звіт до лабораторної роботи.

**Індивідуальне завдання:**

**7.** Клас Stack – стек. Пам’ять під елементи стеку повинна виділятися

динамічно. Реалізувати такі функції-члени:

1. Знаходження максимального значення.
2. Знаходження мінімального значення.
3. Знаходження середнього арифметичного значення стеку.
4. Очищення стеку.
5. Перевірка, чи стек порожній.

Перевантажити операції. При цьому вибір механізму перевантаження

обрати самостійно (чи метод, чи дружня функція):

1. Додавання (почленне додавання елементів до стеку)
2. Віднімання (почленне видалення елементів зі стеку)
3. Множення на скаляр
4. Введення стеку з StringGrid(>>)
5. Виведення стеку StringGrid(<<)
6. Введення стеку з ListBox(>>)
7. Виведення стеку у ListBox(<<)
8. Виведення стеку у Memo(<<)

**Код програми:**

**MyForm.cpp**

#include "MyForm.h"

#include <Windows.h>

using namespace lab07;

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew MyForm);

return 0;

}

**MyForm.h**

#pragma once

#include "HandleStack.h"

#include <iostream>

namespace lab07 {

HandleStack elem;

size\_t counter = 0;

int j = 0;

bool checker = false;

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace std;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::DataGridView^ dataGridView1;

protected:

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::DataGridView^ dataGridView2;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Button^ button3;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::Button^ button4;

private: System::Windows::Forms::Button^ button5;

private: System::Windows::Forms::TextBox^ textBox4;

private: System::Windows::Forms::Button^ button6;

private: System::Windows::Forms::Button^ button7;

private: System::Windows::Forms::Button^ button8;

private: System::Windows::Forms::Button^ button9;

private: System::Windows::Forms::Button^ button10;

private: System::Windows::Forms::ListBox^ listBox1;

private: System::Windows::Forms::ListBox^ listBox2;

private: System::Windows::Forms::Button^ button11;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->dataGridView1 = (gcnew System::Windows::Forms::DataGridView());

this->button1 = (gcnew System::Windows::Forms::Button());

this->label1 = (gcnew System::Windows::Forms::Label());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->dataGridView2 = (gcnew System::Windows::Forms::DataGridView());

this->button2 = (gcnew System::Windows::Forms::Button());

this->label2 = (gcnew System::Windows::Forms::Label());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->button3 = (gcnew System::Windows::Forms::Button());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->button4 = (gcnew System::Windows::Forms::Button());

this->button5 = (gcnew System::Windows::Forms::Button());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->button6 = (gcnew System::Windows::Forms::Button());

this->button7 = (gcnew System::Windows::Forms::Button());

this->button8 = (gcnew System::Windows::Forms::Button());

this->button9 = (gcnew System::Windows::Forms::Button());

this->button10 = (gcnew System::Windows::Forms::Button());

this->listBox1 = (gcnew System::Windows::Forms::ListBox());

this->listBox2 = (gcnew System::Windows::Forms::ListBox());

this->button11 = (gcnew System::Windows::Forms::Button());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView2))->BeginInit();

this->SuspendLayout();

//

// dataGridView1

//

this->dataGridView1->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView1->Location = System::Drawing::Point(12, 12);

this->dataGridView1->Name = L"dataGridView1";

this->dataGridView1->RowHeadersWidth = 51;

this->dataGridView1->RowTemplate->Height = 24;

this->dataGridView1->Size = System::Drawing::Size(376, 190);

this->dataGridView1->TabIndex = 0;

//

// button1

//

this->button1->Enabled = false;

this->button1->Location = System::Drawing::Point(12, 208);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(376, 23);

this->button1->TabIndex = 1;

this->button1->Text = L"Зчитати дані з таблиці";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::button1\_Click);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12));

this->label1->Location = System::Drawing::Point(394, 12);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(216, 25);

this->label1->TabIndex = 2;

this->label1->Text = L"Введіть розмір стеку:";

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(619, 9);

this->textBox1->Multiline = true;

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(52, 35);

this->textBox1->TabIndex = 3;

//

// dataGridView2

//

this->dataGridView2->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView2->Location = System::Drawing::Point(758, 12);

this->dataGridView2->Name = L"dataGridView2";

this->dataGridView2->RowHeadersWidth = 51;

this->dataGridView2->RowTemplate->Height = 24;

this->dataGridView2->Size = System::Drawing::Size(376, 190);

this->dataGridView2->TabIndex = 4;

//

// button2

//

this->button2->Location = System::Drawing::Point(677, 9);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(75, 35);

this->button2->TabIndex = 5;

this->button2->Text = L"OK";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &MyForm::button2\_Click);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12));

this->label2->Location = System::Drawing::Point(394, 64);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(158, 25);

this->label2->TabIndex = 6;

this->label2->Text = L"Введіть скаляр:";

//

// textBox2

//

this->textBox2->Location = System::Drawing::Point(558, 64);

this->textBox2->Multiline = true;

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(64, 35);

this->textBox2->TabIndex = 7;

//

// button3

//

this->button3->Location = System::Drawing::Point(628, 64);

this->button3->Name = L"button3";

this->button3->Size = System::Drawing::Size(124, 35);

this->button3->TabIndex = 8;

this->button3->Text = L"Порахувати";

this->button3->UseVisualStyleBackColor = true;

this->button3->Click += gcnew System::EventHandler(this, &MyForm::button3\_Click);

//

// textBox3

//

this->textBox3->Location = System::Drawing::Point(399, 116);

this->textBox3->Multiline = true;

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(64, 35);

this->textBox3->TabIndex = 9;

//

// button4

//

this->button4->Location = System::Drawing::Point(469, 116);

this->button4->Name = L"button4";

this->button4->Size = System::Drawing::Size(283, 35);

this->button4->TabIndex = 10;

this->button4->Text = L"<- Додати це число до списку";

this->button4->UseVisualStyleBackColor = true;

this->button4->Click += gcnew System::EventHandler(this, &MyForm::button4\_Click);

//

// button5

//

this->button5->Location = System::Drawing::Point(399, 167);

this->button5->Name = L"button5";

this->button5->Size = System::Drawing::Size(353, 35);

this->button5->TabIndex = 11;

this->button5->Text = L"Видалити останнє число зі списку";

this->button5->UseVisualStyleBackColor = true;

this->button5->Click += gcnew System::EventHandler(this, &MyForm::button5\_Click);

//

// textBox4

//

this->textBox4->Location = System::Drawing::Point(399, 208);

this->textBox4->Multiline = true;

this->textBox4->Name = L"textBox4";

this->textBox4->ReadOnly = true;

this->textBox4->Size = System::Drawing::Size(353, 35);

this->textBox4->TabIndex = 12;

//

// button6

//

this->button6->Location = System::Drawing::Point(399, 249);

this->button6->Name = L"button6";

this->button6->Size = System::Drawing::Size(353, 35);

this->button6->TabIndex = 13;

this->button6->Text = L"Знайти максимальне";

this->button6->UseVisualStyleBackColor = true;

this->button6->Click += gcnew System::EventHandler(this, &MyForm::button6\_Click);

//

// button7

//

this->button7->Location = System::Drawing::Point(399, 290);

this->button7->Name = L"button7";

this->button7->Size = System::Drawing::Size(353, 35);

this->button7->TabIndex = 14;

this->button7->Text = L"Знайти мінімальне";

this->button7->UseVisualStyleBackColor = true;

this->button7->Click += gcnew System::EventHandler(this, &MyForm::button7\_Click);

//

// button8

//

this->button8->Location = System::Drawing::Point(399, 331);

this->button8->Name = L"button8";

this->button8->Size = System::Drawing::Size(353, 35);

this->button8->TabIndex = 15;

this->button8->Text = L"Знайти середнє арифметичне";

this->button8->UseVisualStyleBackColor = true;

this->button8->Click += gcnew System::EventHandler(this, &MyForm::button8\_Click);

//

// button9

//

this->button9->Location = System::Drawing::Point(399, 372);

this->button9->Name = L"button9";

this->button9->Size = System::Drawing::Size(353, 35);

this->button9->TabIndex = 16;

this->button9->Text = L"Очистити стек";

this->button9->UseVisualStyleBackColor = true;

this->button9->Click += gcnew System::EventHandler(this, &MyForm::button9\_Click);

//

// button10

//

this->button10->Location = System::Drawing::Point(399, 413);

this->button10->Name = L"button10";

this->button10->Size = System::Drawing::Size(353, 35);

this->button10->TabIndex = 17;

this->button10->Text = L"Перевірити чи стек пустий";

this->button10->UseVisualStyleBackColor = true;

this->button10->Click += gcnew System::EventHandler(this, &MyForm::button10\_Click);

//

// listBox1

//

this->listBox1->FormattingEnabled = true;

this->listBox1->ItemHeight = 16;

this->listBox1->Items->AddRange(gcnew cli::array< System::Object^ >(10) {

L"1", L"2", L"3", L"4", L"5", L"6", L"7", L"8",

L"9", L"10"

});

this->listBox1->Location = System::Drawing::Point(12, 237);

this->listBox1->Name = L"listBox1";

this->listBox1->Size = System::Drawing::Size(376, 164);

this->listBox1->TabIndex = 18;

this->listBox1->SelectedIndexChanged += gcnew System::EventHandler(this, &MyForm::listBox1\_SelectedIndexChanged);

//

// listBox2

//

this->listBox2->FormattingEnabled = true;

this->listBox2->ItemHeight = 16;

this->listBox2->Location = System::Drawing::Point(758, 208);

this->listBox2->Name = L"listBox2";

this->listBox2->Size = System::Drawing::Size(376, 164);

this->listBox2->TabIndex = 19;

//

// button11

//

this->button11->Enabled = false;

this->button11->Location = System::Drawing::Point(12, 407);

this->button11->Name = L"button11";

this->button11->Size = System::Drawing::Size(376, 41);

this->button11->TabIndex = 20;

this->button11->Text = L"Додати число";

this->button11->UseVisualStyleBackColor = true;

this->button11->Click += gcnew System::EventHandler(this, &MyForm::button11\_Click);

//

// richTextBox1

//

this->richTextBox1->Location = System::Drawing::Point(758, 379);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->Size = System::Drawing::Size(376, 69);

this->richTextBox1->TabIndex = 21;

this->richTextBox1->Text = L"";

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(1146, 460);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->button11);

this->Controls->Add(this->listBox2);

this->Controls->Add(this->listBox1);

this->Controls->Add(this->button10);

this->Controls->Add(this->button9);

this->Controls->Add(this->button8);

this->Controls->Add(this->button7);

this->Controls->Add(this->button6);

this->Controls->Add(this->textBox4);

this->Controls->Add(this->button5);

this->Controls->Add(this->button4);

this->Controls->Add(this->textBox3);

this->Controls->Add(this->button3);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->label2);

this->Controls->Add(this->button2);

this->Controls->Add(this->dataGridView2);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->label1);

this->Controls->Add(this->button1);

this->Controls->Add(this->dataGridView1);

this->Name = L"MyForm";

this->Text = L"Stack";

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView2))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

dataGridView1->ColumnCount = Convert::ToInt16(textBox1->Text);

dataGridView1->RowCount = 1;

dataGridView2->ColumnCount = Convert::ToInt16(textBox1->Text);

dataGridView2->RowCount = 1;

int size = Convert::ToInt16(textBox1->Text);

HandleStack elem1(size);

elem = elem1;

button1->Enabled = true;

button11->Enabled = true;

}

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (checker == false)

{

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

{

elem.mass[i] = Convert::ToInt16(dataGridView1->Rows[0]->Cells[i]->Value);

listBox2->Items->Add(Convert::ToString(elem.mass[i]));

richTextBox1->Text = richTextBox1->Text + " " + Convert::ToString(elem.mass[i]);

dataGridView2->Rows[0]->Cells[i]->Value = elem.mass[i];

}

checker = true;

}

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e)

{

elem\* Convert::ToInt16(textBox2->Text);

richTextBox1->Text = "";

listBox2->Items->Clear();

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

{

dataGridView2->Rows[0]->Cells[i]->Value = elem.mass[i];

listBox2->Items->Add(elem.mass[i]);

richTextBox1->Text = richTextBox1->Text + " " + Convert::ToString(elem.mass[i]);

}

}

private: System::Void button6\_Click(System::Object^ sender, System::EventArgs^ e)

{

textBox4->Text = "Найбільше число -> " + elem.FindMax();

}

private: System::Void button7\_Click(System::Object^ sender, System::EventArgs^ e)

{

textBox4->Text = "Найменше число -> " + elem.FindMin();

}

private: System::Void button8\_Click(System::Object^ sender, System::EventArgs^ e)

{

textBox4->Text = "Середнє арифметичне -> " + elem.FindAvg();

}

private: System::Void button9\_Click(System::Object^ sender, System::EventArgs^ e)

{

elem.StackClear();

textBox4->Text = "Стек очищений";

dataGridView1->RowCount = 1;

dataGridView1->ColumnCount = 0;

dataGridView2->RowCount = 1;

dataGridView2->ColumnCount = 0;

richTextBox1->Text = "";

listBox2->Items->Clear();

checker = false;

}

private: System::Void button10\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (elem.IsEmpty())

{

textBox4->Text = "Стек пустий";

}

else

textBox4->Text = "Стек не пустий";

}

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e)

{

int number = Convert::ToInt16(textBox3->Text);

elem.AddNew(number);

dataGridView2->ColumnCount = elem.size - 1;

listBox2->Items->Clear();

richTextBox1->Text = "";

for (size\_t i = 0; i < elem.size - 1; i++)

{

dataGridView2->Rows[0]->Cells[i]->Value = elem.mass[i];

listBox2->Items->Add(elem.mass[i]);

richTextBox1->Text = richTextBox1->Text + " " + Convert::ToString(elem.mass[i]);

}

}

private: System::Void button5\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (!elem.IsEmpty())

{

elem.DeleteLast();

dataGridView2->ColumnCount = elem.size;

listBox2->Items->RemoveAt(elem.size);

richTextBox1->Text = "";

for (size\_t i = 0; i < elem.size - 1; i++)

{

richTextBox1->Text = richTextBox1->Text + " " + Convert::ToString(elem.mass[i]);

dataGridView2->Rows[0]->Cells[i]->Value = elem.mass[i];

}

}

else

textBox4->Text = "Стек пустий";

}

private: System::Void listBox1\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e)

{}

private: System::Void button11\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (counter < elem.size && checker == false)

{

listBox2->Items->Add(listBox1->Text);

dataGridView2->Rows[0]->Cells[j]->Value = listBox1->Text;

richTextBox1->Text = richTextBox1->Text + " " + listBox1->Text;

elem.mass[j] = Convert::ToInt16(listBox1->Text);

j++;

counter++;

}

if (counter == elem.size)

{

checker = true;

}

}

};

}

**Stack.h**

using namespace std;

class HandleStack

{

public:

int\* mass;

int size;

HandleStack();

HandleStack(int);

HandleStack(const HandleStack& ref\_Point);

~HandleStack();

void operator\*(int scalar)

{

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

{

mass[i] \*= scalar;

}

}

int FindMax()

{

int temp = mass[0];

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

{

if (mass[i] > temp)

{

temp = mass[i];

}

}

return temp;

}

int FindMin()

{

int temp = mass[0];

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

{

if (mass[i] < temp)

{

temp = mass[i];

}

}

return temp;

}

double FindAvg()

{

double res = 0;

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

{

res += mass[i];

}

res /= size;

return res;

}

void StackClear()

{

delete[]mass;

size = 0;

}

bool IsEmpty()

{

bool checker = true;

if (size == 0)

{

return checker;

}

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

{

if (mass[i] != 0)

{

checker = false;

break;

}

}

return checker;

}

HandleStack& operator=(HandleStack& st1)

{

size = st1.size;

delete mass;

mass = new int[size];

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

{

mass[i] = st1.mass[i];

}

return \*this;

}

void AddNew(int number)

{

int\* tmp = new int[size + 1];

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

{

tmp[i] = mass[i];

}

size++;

delete[]mass;

mass = new int[size + 1];

for (size\_t i = 0; i < size - 1; i++)

{

mass[i] = tmp[i];

}

mass[size - 1] = number;

delete[]tmp;

}

void DeleteLast()

{

int\* tmp = new int[size + 1];

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

{

tmp[i] = mass[i];

}

size--;

delete[]mass;

mass = new int[size + 1];

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

{

mass[i] = tmp[i];

}

delete[]tmp;

}

};

**Stack.cpp**

#include "HandleStack.h"

HandleStack::HandleStack()

{}

HandleStack::HandleStack(int size)

{

this->size = size;

this->mass = new int[size + 1];

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

{

mass[i] = 0;

}

}

HandleStack::HandleStack(const HandleStack& ref\_Point) //Конструктор копіювання

{

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

{

mass[i] = ref\_Point.mass[i];

}

}

HandleStack::~HandleStack()

{

if (size > 0)

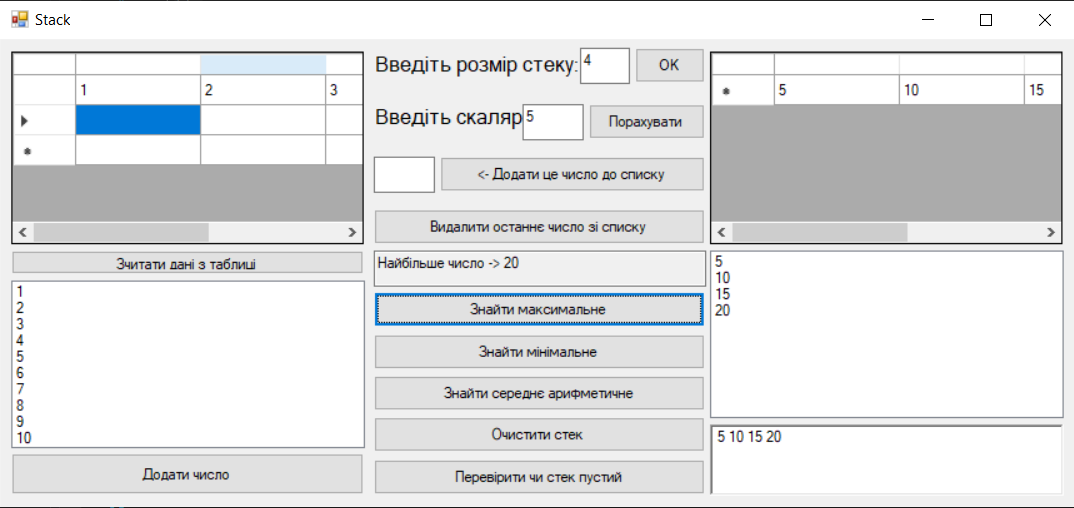
{

delete []mass;

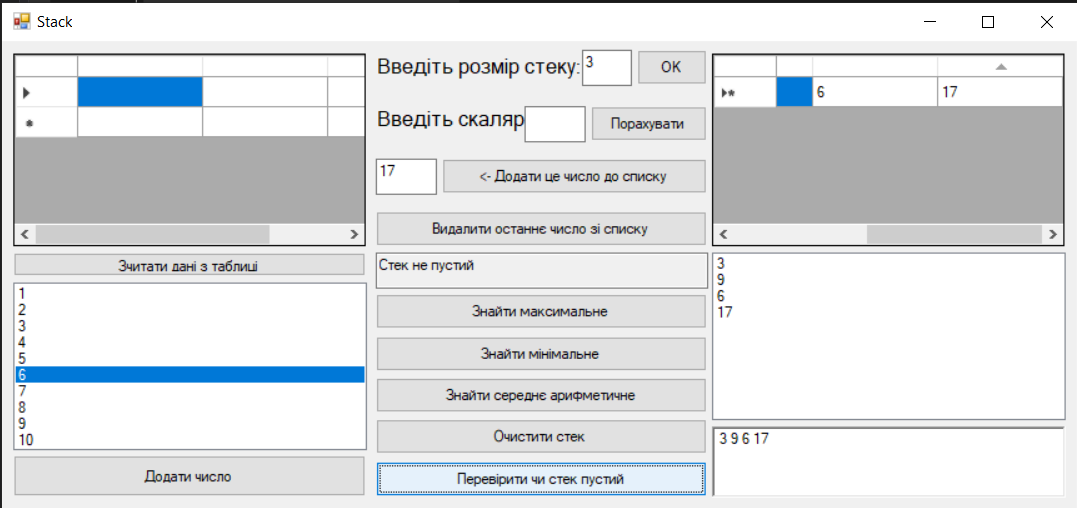
}

}

**Результат виконання:**

****

**Зліва StringGrid, з якої вводили дані та ListBox, в якому ми можемо вивести доступні числа. Справа зверху донизу StringGrid, ListBox та Memo. На даному скріншоті ми перемножили наші введені числа на скаляр та знайшли найбільше число.**

****

**На даному скріншоті ми ввели дані з ListBox та до нашого стеку додали ще один елемент. А також перевірили чи пустий у нас стек.**

**Висновок**

Виконуючи дану лабораторну роботу, я навчивсь використовувати динамічну пам’ять при роботі з класами. Також познайомивсь з функціями new та delete та використав їх у власно розробленій програмі.