МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут **ІКНІ**

Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи №10

На тему: «Шаблони класів»

З дисципліни «Об’єктно-орієнтоване програмування»

Лектор: доцент каф. ПЗ

Коротєєва Т.О.

Виконала: ст.гр. ПЗ-23

Кохман О.В.

Прийняла: доцент каф. ПЗ

Коротєєва Т.О.

«_____»_____2022р.

∑ ____ _____.

Львів – 2022

Тема: шаблони класів.

Мета: навчитись створювати шаблони класів та екземпляри шаблонів.

Теоретичні відомості

У мові C++ **шаблони функцій — це функції, які служать взірцем для створення інших подібних функцій**. Головна ідея — створення функцій без вказівки точного типу(ів) деяких або всіх змінних. Для цього ми визначаємо функцію, вказуючи **тип параметра шаблону**, який використовується замість будь-якого типу даних. Після того, як ми створили функцію з типом параметра шаблону, ми фактично створили «трафарет функції».

Детальне оголошення параметрів шаблону:

Спочатку пишемо **ключове слово template**, яке повідомляє компілятору, що далі ми будемо оголошувати параметри шаблону.

Параметри шаблону функції вказуються в кутових дужках (<>).

Для створення типів параметрів шаблону використовуються **ключові слова typename і class**. В базових випадках використання шаблонів функцій різниці між typename і class немає, тому ви можете вибрати будь-яке з двох. Якщо ви використовуєте ключове слово class, то фактичний тип параметрів не обов'язково повинен бути класом (це може бути змінна фундаментального типу даних, вказівник або щось інше).

Потім даємо назву типу параметра шаблону (зазвичай T).

Шаблони класів працюють точно так же, як і шаблони функцій: компілятор копіює шаблон класу, замінюючи типи параметрів шаблону класу на фактичні (передані) типи даних, а потім компілює цю копію. Якщо у вас є шаблон класу, але ви його не використовуєте, то компілятор не буде його навіть компілювати.

Шаблони класів ідеально підходять для реалізації контейнерних класів, тому що дуже часто таким класам доводиться працювати з різними типами даних, а шаблони дозволяють це організувати в мінімальній кількості коду. Хоча синтаксис трохи потворний, і повідомлення про помилки іноді можуть бути «об'ємними», шаблони класів дійсно є однією з кращих і найбільш корисних конструкцій мови C++.

Індивідуальне завдання

2. Створити шаблон класу Array, який містить однотипні елементи. Шаблон класу повинен давати можливість вивести всі елементи на екран, відсортувати всі елементи в порядку зростання та спадання, а також мінімальний з елементів. Продемонструвати функціонал шаблону на створеному користувацькому типі String – символьна стрічка. Для порівняння стрічок використовувати алфавітний порядок.

Код програми

Назва файлу: CArray.h

```
#ifndef CARRAY_H
#define CARRAY_H
#pragma once
#include "CString.h"
template <typename T>
class CArray {
private:
	T* array;
	int size;
public:
	CArray();
	CArray(T* array, int size);
	~CArray();
	void printArray(System::Windows::Forms::RichTextBox^ richTextBox);
	void sortAscending();
	void sortDescending();
	T min();
};
#endif
template<>
class CArray<CString> {
private:
	CString* array;
	int size;
public:
	CArray() {
		this->size = 0;
		this->array = new CString[size];
	}
	CArray(CString* _array, int size) {
		this->size = size;
		this->array = new CString[size];
		for (int i = 0; i < size; i++) {
			this->array[i] = _array[i];
		}
	}
	void printArray(System::Windows::Forms::RichTextBox^ richTextBox) {
		for (int i = 0; i < size; i++) {
			richTextBox << array[i];
		}
		richTextBox->Text += "\n";
	}
	void sortAscending() {
		int length = size;
		while (length >= 0) {
			for (int i = 1; i < size; i++) {
```

```
                        if (array[i - 1] > array[i]) {
                                CString temp = array[i];
                                array[i] = array[i - 1];
                                array[i - 1] = temp;
                        }
                }
                length--;
        }
    }
    void sortDescending() {
        int length = size;
        while (length >= 0) {
                for (int i = 1; i < size; i++) {
                        if (array[i - 1] < array[i]) {
                                CString temp = array[i];
                                array[i] = array[i - 1];
                                array[i - 1] = temp;
                        }
                }
                length--;
        }
    }
    CString min() {
        CString min = array[0];
        for (int i = 0; i < size; i++) {
                if (array[i] < min) {
                        min = array[i];
                }
        }
        return min;
    }
};
```

Назва файлу: Carray.cpp

```
#include "CArray.h"
template <class T>
CArray<T>::CArray() {
    this->size = 0;
    array = new T[size];
}
template <class T>
CArray<T>::CArray(T* _array, int size) {
    this->size = size;
    array = new T[size];
    for (int i = 0; i < size; i++) {
        this->array[i] = _array[i];
    }
}
template <class T>
CArray<T>::~CArray() {
    delete[] array;
}
template <typename T>
void CArray<T>::printArray(System::Windows::Forms::RichTextBox^ richTextBox) {
    for (int i = 0; i < size; i++) {
        std::string init;
        init += array[i];
        String^ str = gcnew String(init.c_str());
        //richTextBox << this;
        richTextBox->Text += str + " ";
    }
    richTextBox->Text += "\n";
```

```
}
template <typename T>
void CArray<T>::sortAscending() {
	int length = size;
	while (length >= 0) {
		for (int i = 1; i < size; i++) {
			if (array[i - 1] > array[i]) {
				T temp = array[i];
				array[i] = array[i - 1];
				array[i - 1] = temp;
			}
		}
		length--;
	}
}
template <typename T>
void CArray<T>::sortDescending() {
	int length = size;
	while (length >= 0) {
		for (int i = 1; i < size; i++) {
			if (array[i - 1] < array[i]) {
				T temp = array[i];
				array[i] = array[i - 1];
				array[i - 1] = temp;
			}
		}
		length--;
	}
}
template <typename T>
T CArray<T>::min() {
	T min = array[0];
	for (int i = 0; i < size; i++) {
		if (array[i] < min) {
			min = array[i];
		}
	}
	return min;
}
```

Назва файлу: CString.h

```
#ifndef CARRAY_H
#define CARRAY_H
#pragma once
#include "CString.h"
template <typename T>
class CArray {
private:
	T* array;
	int size;
public:
	CArray();
	CArray(T* array, int size);
	~CArray();
	void printArray(System::Windows::Forms::RichTextBox^ richTextBox);
	void sortAscending();
	void sortDescending();
	T min();
};
#endif
template<>
class CArray<CString> {
```

```
private:
	CString* array;
	int size;
public:
	CArray() {
		this->size = 0;
		this->array = new CString[size];
	}
	CArray(CString* _array, int size) {
		this->size = size;
		this->array = new CString[size];
		for (int i = 0; i < size; i++) {
			this->array[i] = _array[i];
		}
	}
	void printArray(System::Windows::Forms::RichTextBox^ richTextBox) {
		for (int i = 0; i < size; i++) {
			richTextBox << array[i];
		}
		richTextBox->Text += "\n";
	}
	void sortAscending() {
		int length = size;
		while (length >= 0) {
			for (int i = 1; i < size; i++) {
				if (array[i - 1] > array[i]) {
					CString temp = array[i];
					array[i] = array[i - 1];
					array[i - 1] = temp;
				}
			}
			length--;
		}
	}
	void sortDescending() {
		int length = size;
		while (length >= 0) {
			for (int i = 1; i < size; i++) {
				if (array[i - 1] < array[i]) {
					CString temp = array[i];
					array[i] = array[i - 1];
					array[i - 1] = temp;
				}
			}
			length--;
		}
	}
	CString min() {
		CString min = array[0];
		for (int i = 0; i < size; i++) {
			if (array[i] < min) {
				min = array[i];
			}
		}
		return min;
	}
};
```

Назва файлу: CString.cpp

```
#include "CArray.h"
template <class T>
CArray<T>::CArray() {
```

```
	this->size = 0;
	array = new T[size];
}
template <class T>
CArray<T>::CArray(T* _array, int size) {
	this->size = size;
	array = new T[size];
	for (int i = 0; i < size; i++) {
		this->array[i] = _array[i];
	}
}
template <class T>
CArray<T>::~CArray() {
	delete[] array;
}
template <typename T>
void CArray<T>::printArray(System::Windows::Forms::RichTextBox^ richTextBox) {
	for (int i = 0; i < size; i++) {
		std::string init;
		init += array[i];
		String^ str = gcnew String(init.c_str());
		//richTextBox << this;
		richTextBox->Text += str + " ";
	}
	richTextBox->Text += "\n";
}
template <typename T>
void CArray<T>::sortAscending() {
	int length = size;
	while (length >= 0) {
		for (int i = 1; i < size; i++) {
			if (array[i - 1] > array[i]) {
				T temp = array[i];
				array[i] = array[i - 1];
				array[i - 1] = temp;
			}
		}
		length--;
	}
}
template <typename T>
void CArray<T>::sortDescending() {
	int length = size;
	while (length >= 0) {
		for (int i = 1; i < size; i++) {
			if (array[i - 1] < array[i]) {
				T temp = array[i];
				array[i] = array[i - 1];
				array[i - 1] = temp;
			}
		}
		length--;
	}
}
template <typename T>
T CArray<T>::min() {
	T min = array[0];
	for (int i = 0; i < size; i++) {
		if (array[i] < min) {
			min = array[i];
		}
	}
	return min;
```

}

Назва файлу: MyForm.h

```
#pragma once
#include "CString.h"
#include "CArray.h"
namespace Final2 {
	using namespace System;
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;
	public ref class MyForm : public System::Windows::Forms::Form
	{
	public:
		MyForm(void)
		{
			InitializeComponent();
		}

	protected:
		~MyForm()
		{
			if (components)
			{
				delete components;
			}
		}
	private: System::Windows::Forms::Button^ button5;
	protected:
	private: System::Windows::Forms::RichTextBox^ richTextBox1;
	private: System::Windows::Forms::Button^ button4;
	private: System::Windows::Forms::Button^ button3;
	private: System::Windows::Forms::Button^ button2;
	private: System::Windows::Forms::Button^ button1;
	private:
		System::ComponentModel::Container ^components;
#pragma region Windows Form Designer generated code

#pragma endregion
		CArray<CString>* object;
	private: System::Void button5_Click(System::Object^ sender,
System::EventArgs^ e) {
		int size = 5;
		CString first = CString(5);
		CString second = CString(6);
		CString third = CString(4);
		CString fourth = CString(5);
		CString fifth = CString(6);
		first.random();
		second.random();
		third.random();
		fourth.random();
		fifth.random();
		CString* _array = new CString[size];
		_array[0] = first;
		_array[1] = second;
		_array[2] = third;
		_array[3] = fourth;
		_array[4] = fifth;
		object = new CArray<CString>(_array, size);
		object->printArray(richTextBox1);
```

```
	}
private: System::Void printArray(System::Object^ sender, System::EventArgs^ e) {
	object->printArray(richTextBox1);
}
private: System::Void sortUp(System::Object^ sender, System::EventArgs^ e) {
	object->sortAscending();
	object->printArray(richTextBox1);
}
private: System::Void sortDown(System::Object^ sender, System::EventArgs^ e) {
	object->sortDescending();
	object->printArray(richTextBox1);
}
private: System::Void printMin(System::Object^ sender, System::EventArgs^ e) {
	richTextBox1->Text += "MIN = ";
	richTextBox1 << object->min();
	richTextBox1->Text += "\n";
}
};
}
```

Назва файлу: MyForm.cpp

```
#include "MyForm.h"
using namespace Final2;
int main() {
	Application::EnableVisualStyles();
	Application::SetCompatibleTextRenderingDefault(false);
	Application::Run(gcnew MyForm());
	return 0;
}
```

Протокол роботи

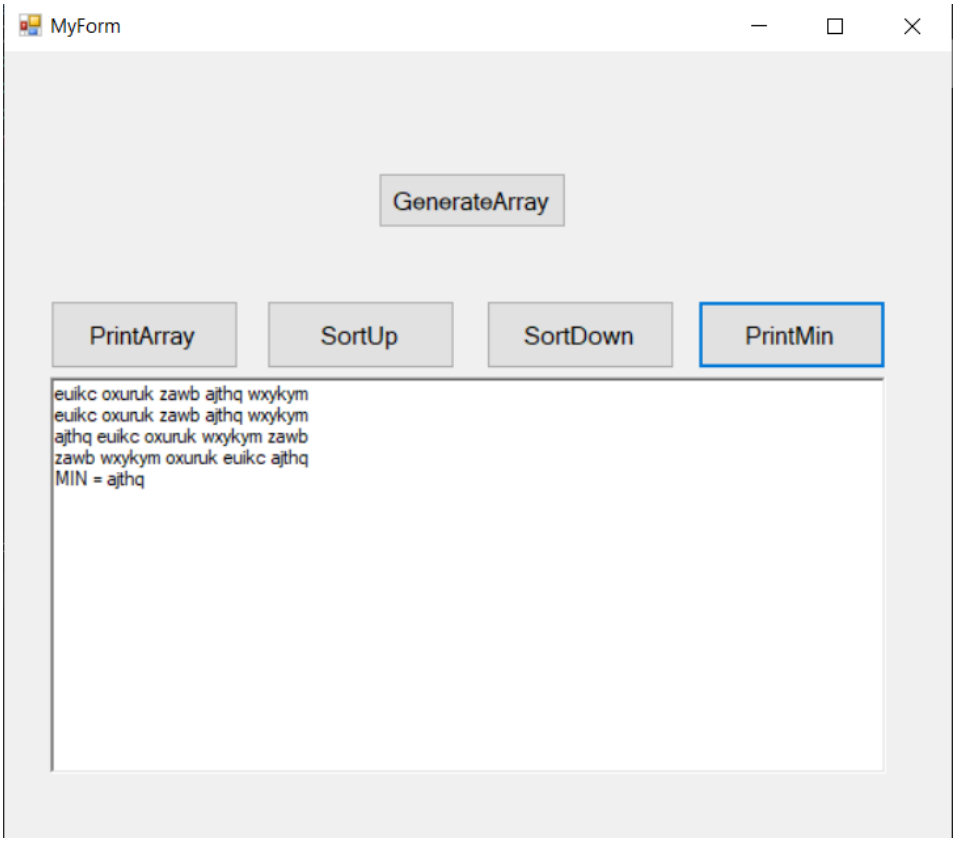

Рис. 1 Результат роботи програми

Висновок

На цій лабораторній роботі я навчилась створювати шаблони класів, використовувати їх та створювати їх екзампляри та продемонструвала результати роботи у консолі в Visual Studio 2022.