Звіт

Автор: Чугунов В.Ю

КІТ-119а

Лабораторна робота №11

**ШАБЛОННІ КЛАСИ**

**Мета:** Поширити знання у шаблонізації на основі вивчення шаблонних класів та створення власних шаблонних типів.

1 ЗАВДАННЯ ДО РОБОТИ

**Загальне завдання.** Модернізувати клас, що був розробленій у попередній роботі наступним шляхом:

* зробити його шаблонним;
* додати поле – шаблонний масив;
* видалити з аргументів існуючих методів масив, а замість цього використовувати масив-поле класу.

2 РОЗРОБКА АЛГОРИТМІВ ФУНКЦІЙ

**Константи:**const int size = 5 – кількість елементів в масивах

**Змінні:**

Template<int, size> ITemp – масив об’єктів шаблонного класу, орієнтований на *int*

int intArr[size] – масив цілих чисел

Template<double, size> FTemp - масив об’єктів шаблонного класу, орієнтований на *double*

double doubleArr[size] – масив реальних чисел

Template<char, size> CTemp - масив об’єктів шаблонного класу, орієнтований на *char*

char charArr[size] – масив символів

Template<Point, size> PTemp - масив об’єктів шаблонного класу, орієнтований на *Point*

Point pointArr[size] – масив об’єктів типу *Point*

**Клас Template<typename T>  
 private поля:**  
T \*arr – покажчик на масив шаблонного типу T

**Функції:  
 void ShowArray ( )** Призначення: виведення елементів масиву на екран.  
 Схему алгоритмів та опис роботи див. у лабораторній роботі №10.

**void SortArray ( )** Призначення: сортування масиву.  
 Схему алгоритмів та опис роботи див. у лабораторній роботі №10.

**int GetIndex ( const T elem )** Призначення: повернення індексу вхідного елементу.  
 Схему алгоритмів та опис роботи див. у лабораторній роботі №10.

**T Min ( )** Призначення: пошук мінімального значення масиву.  
 Схему алгоритмів та опис роботи див. у лабораторній роботі №10.

3 ТЕКСТ ПРОГРАМИ

**Текст файлу main.cpp**#include "Template.h"

#include "Point.h"

int main()

{

const int size = 5;

cout << "#############################################\n";

int intArr[size] = { 5, 84, 2, 46, 35 };

Template<int, size> ITemp = intArr;

cout << "Array of integers:\n";

ITemp.ShowArray();

ITemp.SortArray();

cout << "\nSorted array:\n";

ITemp.ShowArray();

cout << "\nGet index by value: 46 = ";

printf("intArr[%d]\n", ITemp.GetIndex(46));

cout << "Minimal value: " << ITemp.Min() << endl;

cout << "#############################################\n";

double doubleArr[size] = { 8.45, 6.16, 7.95, 3.65, 5.72 };

Template<double, size> FTemp = doubleArr;

cout << "Array of real numbers:\n";

FTemp.ShowArray();

FTemp.SortArray();

cout << "\nSorted array:\n";

FTemp.ShowArray();

cout << "\nGet index by value: 6.49 = ";

printf("floatArr[%d]\n", FTemp.GetIndex(6.49));

cout << "Minimal value: " << FTemp.Min() << endl;

cout << "#############################################\n";

char charArr[size] = "Hola";

Template<char, size> CTemp = charArr;

cout << "Array of chars:\n";

CTemp.ShowArray();

CTemp.SortArray();

cout << "\nSorted array:\n";

CTemp.ShowArray();

cout << "\nGet index by value: o = ";

printf("charArr[%d]\n", CTemp.GetIndex('o'));

cout << "Minimal value: " << CTemp.Min() << endl;

cout << "#############################################\n";

Point pointArr[size] = { {0, 1}, {5, 6}, {3, 4}, {2, 1}, {1, 7} };

Template<Point, size> PTemp = pointArr;

cout << "Array of points:\n";

PTemp.ShowArray();

PTemp.SortArray();

cout << "\nSorted array:\n";

PTemp.ShowArray();

cout << "\nGet index by value: (3 ; 4) = ";

printf("pointArr[%d]\n", PTemp.GetIndex({3, 4}));

cout << "Minimal value: " << PTemp.Min() << endl;

cout << "#############################################\n";

system("PAUSE");

return 0;

}

**Текст файлу Template.h**#pragma once

#include <iostream>

using std::cout;

using std::cin;

using std::endl;

template<typename T, int size>

class Template

{

public:

Template();

Template(const T \*a);

~Template();

void ShowArray();

void SortArray();

int GetIndex(const T elem);

T Min();

private:

T \*arr;

};

template<typename T, int size>

inline Template<T, size>::Template()

{

arr = new T[size];

}

template<typename T, int size>

inline Template<T, size>::Template(const T \*a)

{

arr = new T[size];

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

arr[i] = a[i];

}

template<typename T, int size>

void Template<T, size>::ShowArray()

{

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

cout << arr[i] << '\t';

}

template<typename T, int size>

void Template<T, size>::SortArray()

{

T temp;

for (int i = 1, j; i < size; i++)

{

temp = arr[i];

for (j = i - 1; j >= 0 && arr[j] > temp; j--)

arr[j + 1] = arr[j];

arr[j + 1] = temp;

}

}

template<typename T, int size>

int Template<T, size>::GetIndex(const T elem)

{

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

if (arr[i] == elem)

return i;

return -1;

}

template<typename T, int size>

inline T Template<T, size>::Min()

{

T min = arr[0];

for (int i = 1; i < size; i++)

if (arr[i] < min)

min = arr[i];

return min;

}

template<typename T, int size>

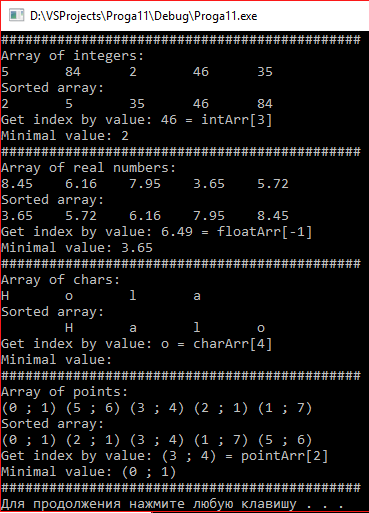
inline Template<T, size>::~Template()

{

delete[] arr;

}

4 РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ

  
Рисунок 1 – Результат роботи програми

ВИСНОВКИ

При виконанні лабораторної роботи набуто практичних навичок щодо розроблення програм з використанням шаблонних класів. Досліджено механізм шаблонізації класів. Він дає можливість використання будь-яких типів даних при створенні об’єкту такого класу.