Автор: Наймитенко С. КІТ-119а

Дата: 15 квітня 2020

# **Лабораторна робота №10**

**ШАБЛОННІ ФУНКЦІЇ**

Мета роботи: отримати базові знання про шаблонізацію (узагальнення) на основі шаблонних функцій.

1 ЗАВДАННЯ ДО РОБОТИ

Створити клас, який не має полів, а всі необхідні дані передаються безпосередньо у функції. Клас має виконувати такі дії: - виводити вміст масиву на екран; - визначати індекс переданого елемента в заданому масиві; - сортувати елементи масиву; - визначати значення мінімального елемента масиву. При цьому необхідно продемонструвати роботу програми як з використанням стандартних типів даних, так і типів, створених користувачем.

**Опис змінних**

Float\* array – масив елементів;

Int min\_value – мінімальний елемент масиву;

Int index – індекс елемента котрий ви хочете повернути;

MyClass obj – об’єкт класу;

Bool flag – флаг для введення вірного індексу елемента;

String name – назва книги;

String author – ім’я автора книги;

Int year\_creating – рік написання книги

Опис функцій

bool Test\_Check\_index() – тест функції перевірки індексу данного елементу;

bool Test\_Min\_value() - визначення мінімального елементу в масиві;

void arrayfloat() – функція для типу float;

Book\* createArray() – створення масиву типу користувача;

float\* Create\_arr() – створення масиву стандартного типу;

T2 Min\_value(T1 array, T2 min\_value) – визначення мінімального значення в масиві;

T1 Sort(T1 array, int choice) – сортуванння масиву;

int Check\_index(T1\* array, T1 value) – перевірка індексу елемента;

void Output\_array(T1 array) – виведення на екран;

**Текст програми**

Файл Base\_Class.h

#pragma once

#include"MyClass.h"

class Book

{

protected:

string name;

string author;

int year\_creating;

public:

Book();

Book(string name, string surname, int age);

Book(const Book& obj);

virtual ~Book();

friend ostream& operator<<(ostream& output, const Book& obj) noexcept;

friend istream& operator>>(istream& input, Book& obj) noexcept;

bool operator==(const Book obj) const noexcept;

bool operator<(const Book obj) const noexcept;

bool operator>(const Book obj) const noexcept;

virtual string getInfo() const;

};

Файл MyClass.h

#pragma once

#define CRTDBG\_MAP\_ALLOC

#include <crtdbg.h>

#define DEBUG\_NEW new(\_NORMAL\_BLOCK, FILE, \_\_LINE)

#include <iostream>

#include <string>

#include <iomanip>

#include <sstream>

using std::ostream;

using std::istream;

using std::string;

using std::cout;

using std::cin;

using std::endl;

using std::setw;

using std::exception;

using std::istringstream;

using std::stringstream;

#define SIZE 5

template <typename T1>

bool A\_more(T1 a, T1 b) { return a > b; }

template <typename T1>

bool B\_more(T1 a, T1 b) { return b > a; }

class MyClass

{

private:

public:

float\* Create\_arr()

{

float\* array = new float[SIZE];

cout << "Заполните масив данными. Пожалуйста, не повторяйтесь." << endl;

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

{

cin >> array[i];

}

int pr;

int j = 0;

do

{

pr = 0;

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

{

if (array[j] == array[i])

{

cout << "Обнаружен елемент который повторяется, введите пожалуйста новый" << endl;

cin >> array[i];

pr = 1;

}

}

j++;

} while (pr == 1);

return array;

}

template <typename T1>

void Output\_array(T1 array)

{

cout << "Вывод масива на экран" << endl;

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

{

cout << setw(3) << array[i];

}

cout << endl;

}

template <typename T1>

int Check\_index(T1\* array, T1 value)

{

int index = SIZE + 1;

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

{

if (array[i] == value)

{

index = i;

i = SIZE;

}

}

if (index == SIZE + 1)

throw 1;

return index;

}

template <typename T1 >

T1 Sort(T1 array, int choice)

{

auto temp = array[0];

bool sort = false;

cout << "Проиходит сортировка масива" << endl;

for (size\_t i = 0; i < SIZE; i++) /\*\*< проганяємо весь масив за допомогою циклу\*/

{

for (size\_t j = 0; j < SIZE; j++)

{

if (choice == 1)

{

sort = A\_more(array[i], array[j]);

}

else if (choice == 2)

{

sort = B\_more(array[i], array[j]);

}

if (sort) /\*\*< перевіряємо відповідність елементів, якщо вона вірна то міняємо елементі місцями\*/

{

temp = array[i];

array[i] = array[j];

array[j] = temp;

}

}

}

return array;

}

template <typename T1, typename T2>

T2 Min\_value(T1 array, T2 min\_value)

{

min\_value = array[0];

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

{

if (min\_value > array[i])

min\_value = array[i];

}

return min\_value;

}

};

Файл Base\_Class.cpp

#include "Base\_Class.h"

Book::Book() : name("TOP\_BOOK"), author("Pushkin"), year\_creating(1999) {}

Book::Book(string name, string author, int year\_creating) : name(name), author(author), year\_creating(year\_creating) {}

Book::Book(const Book& obj) : name(obj.name), author(obj.author), year\_creating(obj.year\_creating) {}

Book:: ~Book() {}

string Book::getInfo() const

{

stringstream temp;

temp << setw(15) << name << setw(20) << author << setw(6) << year\_creating << endl;

return temp.str();

}

ostream& operator<<(ostream& output, const Book& obj) noexcept

{

output.setf(std::ios::left);

output << obj.getInfo();

return output;

}

istream& operator>>(istream& input, Book& obj) noexcept

{

input >> obj.year\_creating;

return input;

}

bool Book::operator==(const Book obj) const noexcept

{

return this->year\_creating == obj.year\_creating;

}

bool Book::operator<(const Book obj) const noexcept

{

return this->year\_creating < obj.year\_creating;

}

bool Book::operator>(const Book obj) const noexcept

{

return this->year\_creating > obj.year\_creating;

}

Файл Test.cpp

#include"MyClass.h"

bool Test\_Check\_index();

bool Test\_Min\_value();

int main()

{

setlocale(LC\_ALL, "Rus");

bool result;

result = Test\_Check\_index();

if (result) /\*\*< перевірка тесту\*/

cout << "Test\_Check\_index пройден успешно!" << endl;

else

cout << "Test\_Check\_index не пройден!" << endl;

result = false;

result = Test\_Min\_value();

if (result) /\*\*< перевірка тесту\*/

cout << "Test\_Min\_value пройден успешно!" << endl;

else

cout << "Test\_Min\_value не пройден!" << endl;

return 0;

}

bool Test\_Check\_index()

{

MyClass obj;

int\* array = new int[3];

array[2] = 1;

if (2 == obj.Check\_index(array, 1))

{

delete[] array;

return true;

}

else

{

delete[] array;

return false;

}

}

bool Test\_Min\_value()

{

MyClass obj;

int\* array = new int[SIZE];

array[0] = 2;

array[1] = 5;

array[2] = 14;

array[3] = 1;

array[4] = 0;

int value = 15;

if (0 == obj.Min\_value(array, value))

{

delete[] array;

return true;

}

else

{

delete[] array;

return false;

}

}

Файл Main.cpp

#include"MyClass.h"

#include"Base\_Class.h"

void arrayfloat()

{

MyClass obj;

float\* array = obj.Create\_arr();

obj.Output\_array(array);

int min\_value = 0;

min\_value = obj.Min\_value(array, min\_value);

cout << "Минимальный елемент в масиве " << min\_value << endl;

float value;

cout << "Введите елемент индекс которого вы хотите узнать" << endl;

int index;

bool flag = true;

while (flag)

{

try

{

cin >> value;

index = obj.Check\_index(array, value);

cout << "Индех елемента который вы хотели узнать " << index << endl;

flag = false;

}

catch (const int value)

{

if (value == 404)

flag = false;

cout << "Такого елемента нет в масиве, поробуйте еще раз" << endl;

}

}

int res;

cout << "Выберте вариацию сортировки, 2 - от меньшего к боьлшему, 1 - от большего к меньшему" << endl;

cin >> res;

array = obj.Sort(array, res);

obj.Output\_array(array);

delete[] array;

}

Book\* createArray()

{

Book\* array = new Book[SIZE];

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

{

if (i == 0)

{

array[i] = Book();

}

else if (i == 1)

{

array[i] = Book("voina\_i\_mir", "Kto\_to", 18);

}

else if (i == 2)

{

array[i] = Book("Elisey", "Leonidovich", 45);

}

else if (i == 3)

{

array[i] = Book("Dmitry", "Aleksandrovich", 23);

}

else if (i == 4)

{

array[i] = Book("Oliver", "Lvovich", 32);

}

}

return array;

}

void arrayclass()

{

MyClass obj;

Book\* array = createArray();

obj.Output\_array(array);

Book min\_value;

min\_value = obj.Min\_value(array, min\_value);

cout << "Минимальный елемент в масиве " << min\_value << endl;

Book value;

cout << "Введите елемент индекс которого вы хотите узнать" << endl;

int index;

bool flag = true;

while (flag)

{

try

{

cin >> value;

index = obj.Check\_index(array, value);

cout << "Индекс елемента который вы хотели узнать " << index << endl;

flag = false;

}

catch (const int value)

{

if (value == 404)

flag = false;

cout << "Такого елемента нет в масиве, поробуйте еще раз" << endl;

}

}

int res;

cout << "Выберте вариацию сортировки, 2 - от меньшего к боьлшему, 1 - от большего к меньшему" << endl;

cin >> res;

array = obj.Sort(array,res);

obj.Output\_array(array);

delete[] array;

}

int main()

{

setlocale(LC\_ALL, "Rus");

arrayfloat();

arrayclass();

return 0;

}

4 РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ

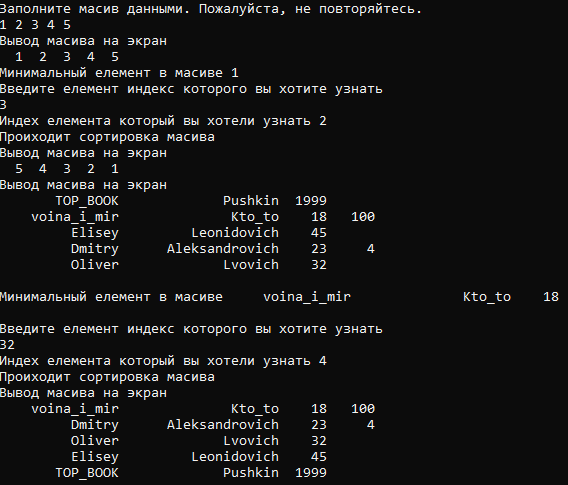


Рисунок 1 - Результат роботи основоного завдання



Рисунок 2 - Результат тестування методів

ВИСНОВКИ

Отримади базові знання про шаблонізацію (узагальнення) на основі шаблонних функцій. Виконали завдання згідно з варіантом 14 у списку.