

Звіт

Автор: Богданов І.Ю. КІТ-119а Дата: 11 травня 2020

Лабораторна робота №10. Шаблонні функції.

Тема. Шаблонні функції.

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

1. Завдання до роботи **Індивідуальне завдання:**

Створити клас, який не має полів, а всі необхідні дані передаються безпосередньо у функції. Пристосувати функції класу, для роботи с масивом довільного типу даних.

2. Опис класів, змінних, методів та функцій

2.1 Опис класів

`class funcclass` – клас призначений для сортування масивів даних різних типів.

`class drob` – клас призначений для демонстрування роботи іншого класа типами даних створених користувачем.

2.2 Опис змінних

`int q` – змінна що визначає з даними якого типу буде працювати програма. Отримується від користувача.

`int n` – змінна що визначає розмір масив. Отримується від користувача.

`int t` – змінна що використовується для реалізації вибору в сторонніх операціях.

`funcclass f` – змінна, що використовується для визова функцій, які є методами класу.

`int* data` – масив типу `int`.

`float* data` – масив типу `float`.

`drob* data` – масив типу `drob`.

`float g` – змінна що використовується для отримання даних типу `float` від користувача.

`drob m` – змінна що використовується для отримання даних типу `drob` від користувача.

2.3 Опис методів

`template <typename t> void print_to_screen(t* data, int l)` – метод класу `funcclass`, виводить на екран зміст даного масиву.

`template <typename t> int find_ind(t* data, int l, t el)` – метод класу `funcclass`, знаходить індекс даного елемента у даному масиві.

`template <typename t> void sort(t* data, int l, int s)` – метод класу `funcclass`, сортує масив у порядку спадання або зростання.

`template <typename t> t find_min(t* data, int l)` – метод класу `funcclass`, знаходити найменший елемент масиву.

`template <typename t> t find_max(t* data, int l)` – метод класу `funcclass`, знаходити найбільший елемент масиву.

`drob& operator=(const drob& d)` – метод класу `drob`, перевантаження оператора присвоювання.

`drob(int f, int s)` – конструктор з параметрами класу `drob`.

`drob()` – конструктор класу `drob`.

2.4 Опис функцій

`std::ostream& operator<<(std::ostream& os, const drob& d)` – перевантаження оператора виведення для класу `drob`.

`std::istream& operator>>(std::istream& is, drob& d)` – перевантаження оператора введення для класу `drob`.

`bool operator==(const drob& d1, const drob& d2)` – перевантаження оператора порівняння для класу `drob`.

`bool operator!=(const drob& d1, const drob& d2)` – перевантаження іншого оператора порівняння для класу `drob`.

`bool operator>(const drob& d1, const drob& d2)` – перевантаження іншого оператора порівняння для класу `drob`.

`bool operator<(const drob& d1, const drob& d2)` – перевантаження іншого оператора порівняння для класу `drob`.

`void menu()` – функція меню.

3 Текст програми

Лабораторная работа 10.cpp

```
#define _CRTDBG_MAP_ALLOC
#include "funcclass.h"

int main(){
    setlocale(LC_ALL, "Russian");
    menu();
    if (_CrtDumpMemoryLeaks()) {
        std::cout << "Утечка памяти обнаружена." << "\n";
    }
    else {
        std::cout << "Утечка памяти не обнаружена." << "\n";
    }
}

funcclass.h
#pragma once
#include <iostream>

class funcclass {
public:
    template <typename t> void print_to_screen(t* data, int l);
    template <typename t> int find_ind(t* data, int l, t el);
    template <typename t> void sort(t* data, int l, int s);
    template <typename t> t find_min(t* data, int l);
    template <typename t> t find_max(t* data, int l);
};

class drob {
public:
    int chislitel;
    int znaminaytel;
```

```

    drob& operator=(const drob& d);
    drob(int f, int s);
    drob();
};
std::ostream& operator<<(std::ostream& os, const drob& d);
std::istream& operator>>(std::istream& is, drob& d);
bool operator==(const drob& d1, const drob& d2);
bool operator!=(const drob& d1, const drob& d2);
bool operator>(const drob& d1, const drob& d2);
bool operator<(const drob& d1, const drob& d2);
void menu();
funcclass.cpp
#include "funcclass.h"

template <typename t> void funcclass::print_to_screen(t* data, int l) {
    std::cout << "Вот ваш массив данных: " << "\n";
    for (int i = 0; i < l; i++) {
        std::cout << i + 1 << " " << data[i] << "\n";
    }
}

template <typename t> int funcclass::find_ind(t* data, int l, t el) {
    for (int i = 0; i < l; i++) {
        if (data[i] == el) {
            return i;
        }
    }
}

template <typename t> void funcclass::sort(t* data, int l, int s) {
    bool check = false;
    if (s >= 0) {
        do {
            check = false;
            for (int i = 0; i < l - 1; i++) {
                if (data[i] < data[i + 1]) {
                    std::swap(data[i], data[i + 1]);
                    check = true;
                }
            }
        } while (check);
    }
    else {
        do {
            check = false;
            for (int i = 0; i < l - 1; i++) {
                if (data[i] > data[i + 1]) {
                    std::swap(data[i], data[i + 1]);
                    check = true;
                }
            }
        } while (check);
    }
}

template <typename t> t funcclass::find_min(t* data, int l) {
    t min = data[0];
    for (int i = 0; i < l; i++) {
        if (min > data[i]) {
            min = data[i];
        }
    }
    return min;
}

template <typename t> t funcclass::find_max(t* data, int l) {
    t max = data[0];
    for (int i = 0; i < l; i++) {
        if (max < data[i]) {
            max = data[i];
        }
    }
    return max;
}

```

```

}
std::ostream& operator<<(std::ostream& os, const drob& d) {
    return os << d.chislitel << "/" << d.znaminaytel;
}
std::istream& operator>>(std::istream& is, drob& d) {
    return is >> d.chislitel >> d.znaminaytel;
}
bool operator==(const drob& d1, const drob& d2) {
    float dr1, dr2;
    dr1 = float(d1.chislitel) / float(d1.znaminaytel);
    dr2 = float(d2.chislitel) / float(d2.znaminaytel);
    if (dr1 == dr2) {
        return true;
    }
    else {
        return false;
    }
}
bool operator!=(const drob& d1, const drob& d2) {
    return !(d1 == d2);
}
bool operator>(const drob& d1, const drob& d2) {
    float dr1, dr2;
    dr1 = float(d1.chislitel) / float(d1.znaminaytel);
    dr2 = float(d2.chislitel) / float(d2.znaminaytel);
    if (dr1 > dr2) {
        return true;
    }
    else {
        return false;
    }
}
bool operator<(const drob& d1, const drob& d2) {
    float dr1, dr2;
    dr1 = float(d1.chislitel) / float(d1.znaminaytel);
    dr2 = float(d2.chislitel) / float(d2.znaminaytel);
    if (dr1 < dr2) {
        return true;
    }
    else {
        return false;
    }
}
drob& drob::operator=(const drob& d) {
    chislitel = d.chislitel;
    znaminaytel = d.znaminaytel;
    return *this;
}
drob::drob(int f, int s) {
    chislitel = f;
    znaminaytel = s;
}
drob::drob() {
    chislitel = 1;
    znaminaytel = 2;
}
void menu() {
    std::cout << "Выберите тип данных который будет использованн в массиве: " << "\n";
    std::cout << "1 - int " << "\n";
    std::cout << "2 - float " << "\n";
    std::cout << "3 - дробное число (кастомный тип данных) " << "\n";
    std::cout << "Введите число соответствующее нужному типу данных: ";
    int q, n, t;
    std::cin >> q;
    funcclass f;
    if (q == 1) {
        int* data;
        std::cout << "Введите размер массива: ";
        std::cin >> n;
    }
}

```

```

data = new int[n];
for (int i = 0; i < n; i++) {
    data[i] = i + 1;
}
f.print_to_screen(data, n);
while (true){
    std::cout << "Выберите нужное действие: " << "\n";
    std::cout << "1 - отсортировать массив" << "\n";
    std::cout << "2 - найти минимум или максимум" << "\n";
    std::cout << "3 - получить индекс элемента по его значению" << "\n";
    std::cout << "4 - показать массив" << "\n";
    std::cout << "5 - завершить работу программы" << "\n";
    std::cout << "Введите число соответствующее нужной операции: ";
    std::cin >> q;
    if (q == 1) {
        std::cout << "Выберите направление сортировки 1 - по возрастанию, 2 - по
убыванию: ";

        std::cin >> t;
        if (t == 1) {
            f.sort(data, n, -1);
        }
        else if (t == 2) {
            f.sort(data, n, 1);
        }
        else {
            std::cout << "Введите правильное значение" << "\n";
        }
    }
    else if (q == 2) {
        std::cout << "Выберите, что будете искать 1 - минимум, 2 - максимум: ";
        std::cin >> t;
        if (t == 1) {
            std::cout << "Ваш минимум: " << f.find_min(data, n) << "\n";
        }
        else if (t == 2) {
            std::cout << "Ваш максимум: " << f.find_max(data, n) << "\n";
        }
        else {
            std::cout << "Введите правильное значение" << "\n";
        }
    }
    else if (q == 3) {
        std::cout << "Введите нужный элемент: ";
        std::cin >> t;
        std::cout << "Индекс вашего элемента: " << f.find_ind(data, n, t) <<
"\n";
    }
    else if (q == 4) {
        f.print_to_screen(data, n);
    }
    else {
        delete[] data;
        break;
    }
}
}
else if (q == 2) {
    float* data;
    float g;
    std::cout << "Введите размер массива: ";
    std::cin >> n;
    data = new float[n];
    for (int i = 0; i < n; i++) {
        data[i] = i + 1 + ((i + 1) / 10);
    }
    f.print_to_screen(data, n);
    while (true) {
        std::cout << "Выберите нужное действие: " << "\n";
        std::cout << "1 - отсортировать массив" << "\n";

```

```

std::cout << "2 - найти минимум или максимум" << "\n";
std::cout << "3 - получить индекс элемента по его значению" << "\n";
std::cout << "4 - показать массив" << "\n";
std::cout << "5 - завершить работу программы" << "\n";
std::cout << "Введите число соответствующее нужной операции: ";
std::cin >> q;
if (q == 1) {
    std::cout << "Выберите направление сортировки 1 - по возрастанию, 2 - по
убыванию: ";

    std::cin >> t;
    if (t == 1) {
        f.sort(data, n, -1);
    }
    else if (t == 2) {
        f.sort(data, n, 1);
    }
    else {
        std::cout << "Введите правильное значение" << "\n";
    }
}
else if (q == 2) {
    std::cout << "Выберите, что будете искать 1 - минимум, 2 - максимум: ";
    std::cin >> t;
    if (t == 1) {
        std::cout << "Ваш минимум: " << f.find_min(data, n) << "\n";
    }
    else if (t == 2) {
        std::cout << "Ваш максимум: " << f.find_max(data, n) << "\n";
    }
    else {
        std::cout << "Введите правильное значение" << "\n";
    }
}
else if (q == 3) {
    std::cout << "Введите нужный элемент: ";
    std::cin >> g;
    std::cout << "Индекс вашего элемента: " << f.find_ind(data, n, g) <<
"\n";

}
else if (q == 4) {
    f.print_to_screen(data, n);
}
else {
    delete[] data;
    break;
}
}
}
else if (q == 3) {
    drob* data;
    drob m;
    std::cout << "Введите размер массива: ";
    std::cin >> n;
    data = new drob[n];
    for (int i = 0; i < n; i++) {
        data[i] = drob(i + 1, i + 2);
    }
    f.print_to_screen(data, n);
    while (true) {
        std::cout << "Выберите нужное действие: " << "\n";
        std::cout << "1 - отсортировать массив" << "\n";
        std::cout << "2 - найти минимум или максимум" << "\n";
        std::cout << "3 - получить индекс элемента по его значению" << "\n";
        std::cout << "4 - показать массив" << "\n";
        std::cout << "5 - завершить работу программы" << "\n";
        std::cout << "Введите число соответствующее нужной операции: ";
        std::cin >> q;
        if (q == 1) {

```

```

        std::cout << "Выберите направление сортировки 1 - по возрастанию, 2 - по
убыванию: ";

        std::cin >> t;
        if (t == 1) {
            f.sort(data, n, -1);
        }
        else if (t == 2) {
            f.sort(data, n, 1);
        }
        else {
            std::cout << "Введите правильное значение" << "\n";
        }
    }
    else if (q == 2) {
        std::cout << "Выберите, что будете искать 1 - минимум, 2 - максимум: ";
        std::cin >> t;
        if (t == 1) {
            std::cout << "Ваш минимум: " << f.find_min(data, n) << "\n";
        }
        else if (t == 2) {
            std::cout << "Ваш максимум: " << f.find_max(data, n) << "\n";
        }
        else {
            std::cout << "Введите правильное значение" << "\n";
        }
    }
    else if (q == 3) {
        std::cout << "Введите нужный элемент, сначала введите числитель в виде
целого числа, а потом знаменатель в таком же виде: ";
        std::cin >> m;
        std::cout << "Индекс вашего элемента: " << f.find_ind(data, n, m) <<
"\n";
    }
    else if (q == 4) {
        f.print_to_screen(data, n);
    }
    else {
        delete[] data;
        break;
    }
}

}

}

tests.cpp
#define _CRTDBG_MAP_ALLOC
#include "funcclass.h"

int main() {
    setlocale(LC_ALL, "Russian");
    funcclass f;
    int* data;
    std::cout << "Если сейчас дважды будет выведен массив в разном порядке, то функции сортировки
и вывода работают правильно." << "\n";
    int n = 10;
    data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = i + 1;
    }
    f.sort(data, n, 1);
    f.print_to_screen(data, n);
    f.sort(data, n, -1);
    f.print_to_screen(data, n);
    delete[] data;
    if (_CrtDumpMemoryLeaks()) {
        std::cout << "Утечка памяти обнаружена." << "\n";
    }
    else {
        std::cout << "Утечка памяти не обнаружена." << "\n";
    }
}

```

```
std::cout << "Нажмите любую клавишу для завершения теста." << "\n";  
std::cin >> n;
```

```
}
```

4. Результаты работы программы

Результаты работы программы:

```
Выберите тип данных который будет использованн в массиве:  
1 - int  
2 - float  
3 - дробное число (кастомный тип данных)  
Введите число соответствующее нужному типу данных: 1  
Введите размер массива: 10  
Вот ваш массив данных:  
1 1  
2 2  
3 3  
4 4  
5 5  
6 6  
7 7  
8 8  
9 9  
10 10  
Выберите нужное действие:  
1 - отсортировать массив  
2 - найти минимум или максимум  
3 - получить индекс элемента по его значению  
4 - показать массив  
5 - завершить работу программы  
Введите число соответствующее нужной операции:
```

Результаты тестів:

```
Если сейчас дважды будет выведен массив в разном порядке, то функции сортировки и вывода работают правильно.  
Вот ваш массив данных:  
1 10  
2 9  
3 8  
4 7  
5 6  
6 5  
7 4  
8 3  
9 2  
10 1  
Вот ваш массив данных:  
1 1  
2 2  
3 3  
4 4  
5 5  
6 6  
7 7  
8 8  
9 9  
10 10  
Утечка памяти не обнаружена.  
Нажмите любую клавишу для завершения теста.
```


5. Висновки

При виконанні даної лабораторної роботи було використано механізм шаблонних функцій, що дозволяє використовувати ту й саму функцію для даних різних типів. Це було досягнуто завдяки використанню ключового слова `template`, що дозволило використати під час написання функції неіснуючий тип даних, що замінюється потрібним типом при виклику цієї функції.

Програма протестована, витоків пам'яті немає, виконується без помилок.