ЛАБОРАТОРНА РОБОТА №7

Тема: Шаблони функцій і класів.
Мета: Навчитись створювати і використовувати шаблонні функції і класи

Завдання 1. Напишіть шаблон функції, що повертає середнє арифметичне всіх елементів масиву. Аргументами функції повинні бути ім'я і розмір масиву (типу int). У функції main() перевірте роботу шаблонної функції з масивами типу int, long, double і char.

Завдання 2. Створіть функцію amax(), що повертає значення максимального елемента масиву. Аргументами функції повинні бути адреса і розмір масиву. Зробіть з функції шаблон, щоб вона могла працювати з масивом будь-якого числового типу. Напишіть функцію main(), в якій перевірите роботу функції з різними типами масивів.

Завдання 3. Створіть шаблонний клас, який міститиме як атрибут - масив будь-якого числового типу. Розмір масиву необхідно визначати параметром конструктора класу. З допомогою методів класу потрібно:

- заповнювати масив;
- виводити значення масиву на екран;
- визначати і вивести середнє арифметичне всіх елементів масиву;
- визначати і вивести максимальний елемент масиву.

Напишіть функцію main(), в якій перевірите роботу класу з різними вбудованими типами даних.

Код програми:

```
#include <iostream>
#include <stdlib.h>
using namespace std;

template <typename T>
double average(T arr[], int capacity) {
    double avg = 0;
    for (int i = 0; i < capacity; i++) {
        avg += arr[i];
    }
    return avg / capacity;
}

template <typename T>
T amax(T arr[], int capacity) {
    T res = arr[0];
    for (int i = 1; i <= capacity; i++) {
        if (res < arr[i]) {
            res = arr[i];
        }
    }
    return res;
}
```

```
template <class T>
class Nums {
private:
    T* arr;
    int capacity;
public:
    Nums() : capacity(0), arr(nullptr) {}
    Nums(int nCap, T *a = 0) {
        arr = new T[nCap];
        capacity = nCap;
        if (a) {
            for (int i = 0; i < nCap; i++) {
                this->arr[i] = a[i];
            }
        }
    }
    ~Nums() {
        if (arr)
            delete[] arr;
    }
    void setCapacity(int nCap) {
        this->capacity = nCap;
    }

    int copyArray(T* array, int len) {
        if (this->capacity <= 0) {
            return EXIT_FAILURE;
        }
        if (this->arr) {
            delete[] this->arr;
        }
        for (int i = 0; i < len; i++) {
            this->arr[i] = array[i];
        }
        return EXIT_SUCCESS;
    }

    int setArray(T* array) {
        if (this->capacity <= 0) {
            return EXIT_FAILURE;
        }
        if (this->arr) {
            delete[] this->arr;
        }
        this->arr = array;
        return EXIT_SUCCESS;
    }

    int setArray() {
        if (this->capacity <= 0) {
            return EXIT_FAILURE;
        }
        if (this->arr) {
            delete[] this->arr;
            this->capacity = 0;
        }
        for (int i = 0; i < this->capacity; i++) {
            cout << "Enter element: ";
            cin >> arr[i];
        }
        return EXIT_SUCCESS;
    }
    double average(T arr[], int capacity) {
        double avg = 0;
        for (int i = 0; i < capacity; i++) {
```

```
            avg += arr[i];
        }
        return avg / capacity;
    }
    T* getArr() const {
        return this->arr;
    }

    void printArray() const {
        int i;
        for (i = 0; i < this->capacity - 1; i++) {
            cout << this->arr[i] << ", ";
        }
        cout << this->arr[i];
        cout << endl;
    }

};

int main()
{
    /****************** task 1 /******************/

    int ai[] = {4, 5, 6, 2};
    long al[] = { 10, 20, 30, 40 };
    double ad[] = { 32.2, 35.7, 22.3 };
    char ch[] = { 'a', 'b', 'c' }; // 97 98 99
    cout << "Average of array is " << average(ai, sizeof(ai) / sizeof(int)) << endl;
    cout << "Average of array is " << average(al, sizeof(al) / sizeof(long)) << endl;
    cout << "Average of array is " << average(ad, sizeof(ad) / sizeof(double)) <<
endl;
    cout << "Average of array is " << average(ch, sizeof(ch) / sizeof(char)) << endl;

    /****************** task 1 /******************/


    /****************** task 2 /******************/

    cout << endl << "MAX value of array is " << amax(ai, sizeof(ai) / sizeof(int)) <<
endl;
    cout << "MAX value of array is " << amax(al, sizeof(al) / sizeof(long)) << endl;
    cout << "MAX value of array is " << amax(ad, sizeof(ad) / sizeof(double)) << endl;
    cout << "MAX value of array is " << amax(ch, sizeof(ch) / sizeof(char)) << endl;

    /****************** task 2 /******************/


    /****************** task 3 /******************/

    cout << endl;

    int lenArrInt = sizeof(ai) / sizeof(int);
    int lenArrchar = sizeof(ch) / sizeof(char);

    Nums<int> arrInt(lenArrInt, ai);
    Nums<char> arrChar(lenArrchar, ch);

    arrInt.printArray();
    cout << average(arrInt.getArr(), lenArrInt) << endl;
    cout << amax(arrInt.getArr(), lenArrInt) << endl << endl;

    arrChar.printArray();
    cout << (char)average(arrChar.getArr(), lenArrchar) << endl;
    cout << amax(arrChar.getArr(), lenArrchar) << endl;
```

```
    /****************** task 3 /******************/

    return 0;
}
```

Результат:

```
Average of array is 4.25
Average of array is 25
Average of array is 30.0667
Average of array is 98

MAX value of array is 6
MAX value of array is 40
MAX value of array is 35.7
MAX value of array is c

4, 5, 6, 2
4.25
6

a, b, c
b
c
```

Індивідуальне завдання:
Варіант 10. Створити шаблонний клас – стек на основі зв’язного списку у динамічній пам’яті. Тип елементів стеку визначається параметром шаблона. Передбачити функції для виконання таких операцій: занесення елемента у стек, вилучення значення з вершини стеку, виведення усіх значень стеку на екран, визначення кількості елементів стеку.

Код програми:

Stack.h:

```
#pragma once
#include <iostream>

template <class T>
class Stack
{
public:
	Stack();
	//Stack() : top(nullptr), size(0)	{}
	~Stack();

	void push(T data);
	T pop();
	int getSize() const;
	void printAll() const;

	class Node {
	public:
		Node(T nData = 0, Node* ptrNext = nullptr);
		Node* next;
		T data;
```

```
	};
private:
	Node* top;
	int size;
};
```

Stack.cpp:

```
#include "Stack.h"
#include <iostream>

using namespace std;

template <class T>
Stack<T>::Stack() : top(nullptr), size(0) {}

template <class T>
Stack<T>::~Stack() {}

template <class T>
void Stack<T>::push(T data) {
	if (top) {                                    // if top != nullptr
		Node* nElem = new Node(data, top);
		top = nElem;
	}
	else {
		this->top = new Node(data);
	}
	this->size++;
}

template <class T>
T Stack<T>::pop() {
	if (top != nullptr) {
		Node* tmp = top;
		int res = top->data;
		top = tmp->next;
		delete tmp;
		--size;
		return res;
	}
	else {
		std::cout << "empty stack" << std::endl;
		return 0;
	}
}

template <class T>
int Stack<T>::getSize() const {
	return this->size;
}

template <class T>
Stack<T>::Node::Node(T nData, Node* ptrNext) : data(nData), next(ptrNext) {}

template <class T>
void Stack<T>::printAll() const {
	Node* curr = top;
	while (curr != nullptr) {
		std::cout << curr->data << std::endl;
		curr = curr->next;
	}
}
```

Результат:

```
current size:1
current size:2
8
16
popped 8
current size:1
16
popped 16
```

Висновок: навчився створювати і використовувати шаблонні функції і класи.