Міністерство освіти і науки України

Відокремлений структурний підрозділ «Тернопільський фаховий коледж Тернопільського національного технічного університету імені Івана Пулюя»

*Циклова комісія комп’ютерних наук*

# ЗВІТ

про виконання лабораторної роботи №7

з дисципліни: “Об’єктивно-орієнтоване програмування”

на тему: “Шаблони функцій і класів.”

Виконав:

студент групи КН-323

Руда. Л. І.

Прийняв:

Слободян Р. О.

### Тернопіль 2024

ЛАБОРАТОРНА РОБОТА № 7

Мета: Навчитись створювати і використовувати шаблонні функції і класи

ПОРЯДОК ВИКОНАННЯ РОБОТИ

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 аmaх(), 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.

Завдання 4. Напишіть програму згідно індивідуального завдання.

ІНДИВІДУАЛЬНІ ЗАВДАННЯ.

Створити шаблонний клас – однозв’язний лінійний список у

динамічній пам’яті. Тип елементів списку визначається параметром шаблона.

Передбачити функції для виконання таких операцій: створення нового елемента

списку на його початку, вилучення першого елемента списку, створення нового

елемента списку у його кінці, вилучення останнього елемента списку, виведення

списку на екран.

Завдання 1

**#include** <iostream>

**#include** <numeric> // для accumulate

**using** **namespace** std;

**template** <**typename** T>

**double** average(**const** T**\*** array, **int** size) {

T sum **=** std::accumulate(array, array **+** size, T(0));

**return** **static\_cast<double>**(sum) **/** size;

}

**int** main() {

// int

**int** intArray[] **=** {20, 28, 13, 22, 15};

**int** intSize **=** **sizeof**(intArray) **/** **sizeof**(intArray[0]);

cout **<<** "Average of int array: " **<<** average(intArray, intSize) **<<** endl;

// long

**long** longArray[] **=** {141414, 202020**L**, 111111**L**, 313131**L**, 616161**L**};

**int** longSize **=** **sizeof**(longArray) **/** **sizeof**(longArray[0]);

cout **<<** "Average of long array: " **<<** average(longArray, longSize) **<<** endl;

// double

**double** doubleArray[] **=** {15.5, 154.5, 19.5, 4.5, 31.5};

**int** doubleSize **=** **sizeof**(doubleArray) **/** **sizeof**(doubleArray[0]);

cout **<<** "Average of double array: " **<<** average(doubleArray, doubleSize) **<<** endl;

// char

**char** charArray[] **=** {100, 110, 115, 120, 125};

**int** charSize **=** **sizeof**(charArray) **/** **sizeof**(charArray[0]);

cout **<<** "Average of char array: " **<<** average(charArray, charSize) **<<** endl;

**return** 0;

}

Завдання 2

**#include** <iostream>

**#include** <algorithm> // для std::max

**using** **namespace** std;

// Шаблон функції для знаходження максимального елемента

**template** <**typename** T>

T amax(**const** T**\*** array, **int** size) {

**if** (size **<=** 0) {

std::cerr **<<** "Розмір масиву має бути більшим за ноль" **<<** std::endl;

**return** T(); // Повертаємо нейтральний елемент

}

T max\_value **=** array[0];

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

**if** (array[i] **>** max\_value) {

max\_value **=** array[i];

}

}

**return** **\***max\_element(array, array **+** size);

}

**int** main() {

// Тест для масиву типу int

**int** intArray[] **=** {20, 28, 13, 22, 15};

**int** intSize **=** **sizeof**(intArray) **/** **sizeof**(intArray[0]);

cout **<<** "Max of int array: " **<<** amax(intArray, intSize) **<<** endl;

// Тест для масиву типу long

**long** longArray[] **=** {141414, 202020**L**, 111111**L**, 313131**L**, 616161**L**};

**int** longSize **=** **sizeof**(longArray) **/** **sizeof**(longArray[0]);

cout **<<** "Max of long array: " **<<** amax(longArray, longSize) **<<** endl;

// Тест для масиву типу double

**double** doubleArray[] **=** {15.5, 154.5, 19.5, 4.5, 31.5};

**int** doubleSize **=** **sizeof**(doubleArray) **/** **sizeof**(doubleArray[0]);

cout **<<** "Max of double array: " **<<** amax(doubleArray, doubleSize) **<<** endl;

// Тест для масиву типу float

**float** floatArray[] **=** {3.14**f**, 2.71**f**, 1.61**f**, 4.2**f**, 2.8**f**};

**int** floatSize **=** **sizeof**(floatArray) **/** **sizeof**(floatArray[0]);

cout **<<** "Max of float array: " **<<** amax(floatArray, floatSize) **<<** endl;

**return** 0;

}

Завдання 3

**#include** <iostream>

**#include** <algorithm> // max\_element

**#include** <numeric> // accumulate

**#include** <ctime> // для генерації випадкових чисел

**#include** <cstdlib> // для rand()

**using** **namespace** std;

**template** <**typename** T>

**class** Array {

**private:**

T**\*** data; // Вказівник на масив

**int** size; // Розмір

**public:**

// Конструктор для виділення пам'яті під масив

Array(**int** size) : size(size) {

data **=** **new** T[size];

}

// Деструктор для очищення пам'яті

~Array() {

**delete[]** data;

}

// Заповнення масиву

**void** fillArray() {

cout **<<** "Enter " **<<** size **<<** " elements: ";

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

cin **>>** data[i];

}

}

// Автоматичне заповнення масиву

**void** fillRandom() {

srand(**static\_cast<unsigned>**(time(0))); // Ініціалізація генератора випадкових чисел

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

data[i] **=** **static\_cast<**T**>**(rand() **%** 100); // Генерація чисел від 0 до 99

}

}

// Виведення на екран

**void** printArray() **const** {

cout **<<** "Array elements: ";

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

cout **<<** data[i] **<<** " ";

}

cout **<<** endl;

}

// Середнє арифметичне

**double** calculateAverage() **const** {

T sum **=** std::accumulate(data, data **+** size, T(0));

**return** **static\_cast<double>**(sum) **/** size;

}

// Максимальний елемент

T findMax() **const** {

**return** **\***std::max\_element(data, data **+** size);

}

};

**int** main() {

// Масив int

Array**<int>** intArray(5);

intArray.fillRandom();

intArray.printArray();

cout **<<** "Average of int array: " **<<** intArray.calculateAverage() **<<** endl;

cout **<<** "Max of int array: " **<<** intArray.findMax() **<<** endl;

// Масив double

Array**<double>** doubleArray(5);

doubleArray.fillArray();

doubleArray.printArray();

cout **<<** "Average of double array: " **<<** doubleArray.calculateAverage() **<<** endl;

cout **<<** "Max of double array: " **<<** doubleArray.findMax() **<<** endl;

**return** 0;

}

Завдання 4

**#include** <iostream>

**using** **namespace** std;

// Шаблонна структура для елемента списку

**template** <**typename** T>

**struct** Node {

T data; // Дані елемента

Node**\*** next; // Вказівник на наступний елемент

// Конструктор

Node(T value) : data(value), next(**nullptr**) {}

};

// Шаблонний клас для однозв'язного лінійного списку

**template** <**typename** T>

**class** SinglyLinkedList {

**private:**

Node**<**T**>\*** head; // Вказівник на перший елемент списку

**public:**

// Конструктор

SinglyLinkedList() : head(**nullptr**) {}

// Деструктор

~SinglyLinkedList() {

clear(); // Очищення списку при знищенні

}

// Метод для очищення списку

**void** clear() {

**while** (head) {

removeFirst(); // Видалення всіх елементів

}

}

// Додавання нового елемента на початок списку

**void** addFirst(T value) {

Node**<**T**>\*** newNode **=** **new** Node<T>(value);

newNode->next **=** head;

head **=** newNode;

}

// Вилучення першого елемента списку

**void** removeFirst() {

**if** (**!**head) {

cout **<<** "List is empty. Cannot remove first element." **<<** endl;

**return**;

}

Node**<**T**>\*** temp **=** head;

head **=** head->next;

**delete** temp;

}

// Додавання нового елемента в кінець списку

**void** addLast(T value) {

Node**<**T**>\*** newNode **=** **new** Node<T>(value);

**if** (**!**head) {

head **=** newNode; // Якщо список порожній, новий елемент стає головою

**return**;

}

Node**<**T**>\*** current **=** head;

**while** (current->next) {

current **=** current->next; // Знайти останній елемент

}

current->next **=** newNode; // Додати новий елемент в кінець

}

// Вилучення останнього елемента списку

**void** removeLast() {

**if** (**!**head) {

cout **<<** "List is empty. Cannot remove last element." **<<** endl;

**return**;

}

**if** (**!**head->next) {

**delete** head; // Якщо в списку лише один елемент

head **=** **nullptr**;

**return**;

}

Node**<**T**>\*** current **=** head;

**while** (current->next->next) {

current **=** current->next; // Знайти передостанній елемент

}

**delete** current->next; // Видалити останній елемент

current->next **=** **nullptr**; // Встановити вказівник на null

}

// Виведення списку на екран

**void** printList() **const** {

Node**<**T**>\*** current **=** head;

cout **<<** "List elements: ";

**while** (current) {

cout **<<** current->data **<<** " ";

current **=** current->next;

}

cout **<<** endl;

}

};

**int** main() {

// Використання списку з типом int

SinglyLinkedList**<int>** intList;

**int** number;

**char** choice;

**do** {

cout **<<** "Enter a number to add to the list: ";

cin **>>** number;

intList.addLast(number); // Додаємо число в кінець списку

cout **<<** "Do you want to add another number? (y/n): ";

cin **>>** choice;

} **while** (choice **==** 'y' **||** choice **==** 'Y');

intList.printList(); // Виведе всі елементи списку

// Вилучення елементів

**char** removeChoice;

**do** {

cout **<<** "Do you want to remove the first element? (y/n): ";

cin **>>** removeChoice;

**if** (removeChoice **==** 'y' **||** removeChoice **==** 'Y') {

intList.removeFirst();

intList.printList(); // Виведе всі елементи списку

}

} **while** (removeChoice **==** 'y' **||** removeChoice **==** 'Y');

// Вилучення останнього елемента

**char** removeLastChoice;

cout **<<** "Do you want to remove the last element? (y/n): ";

cin **>>** removeLastChoice;

**if** (removeLastChoice **==** 'y' **||** removeLastChoice **==** 'Y') {

intList.removeLast();

intList.printList(); // Виведе всі елементи списку

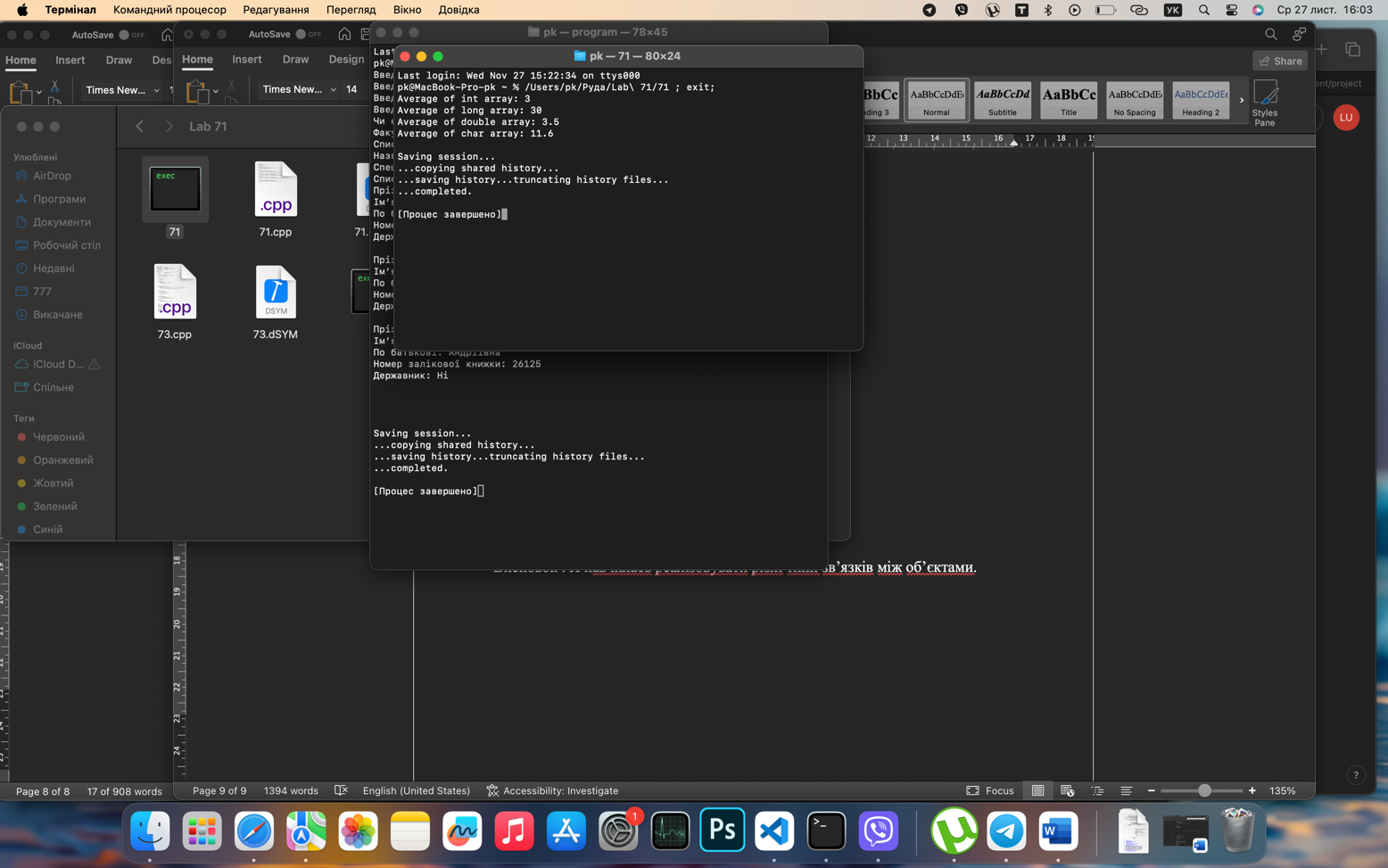
}

**return** 0;

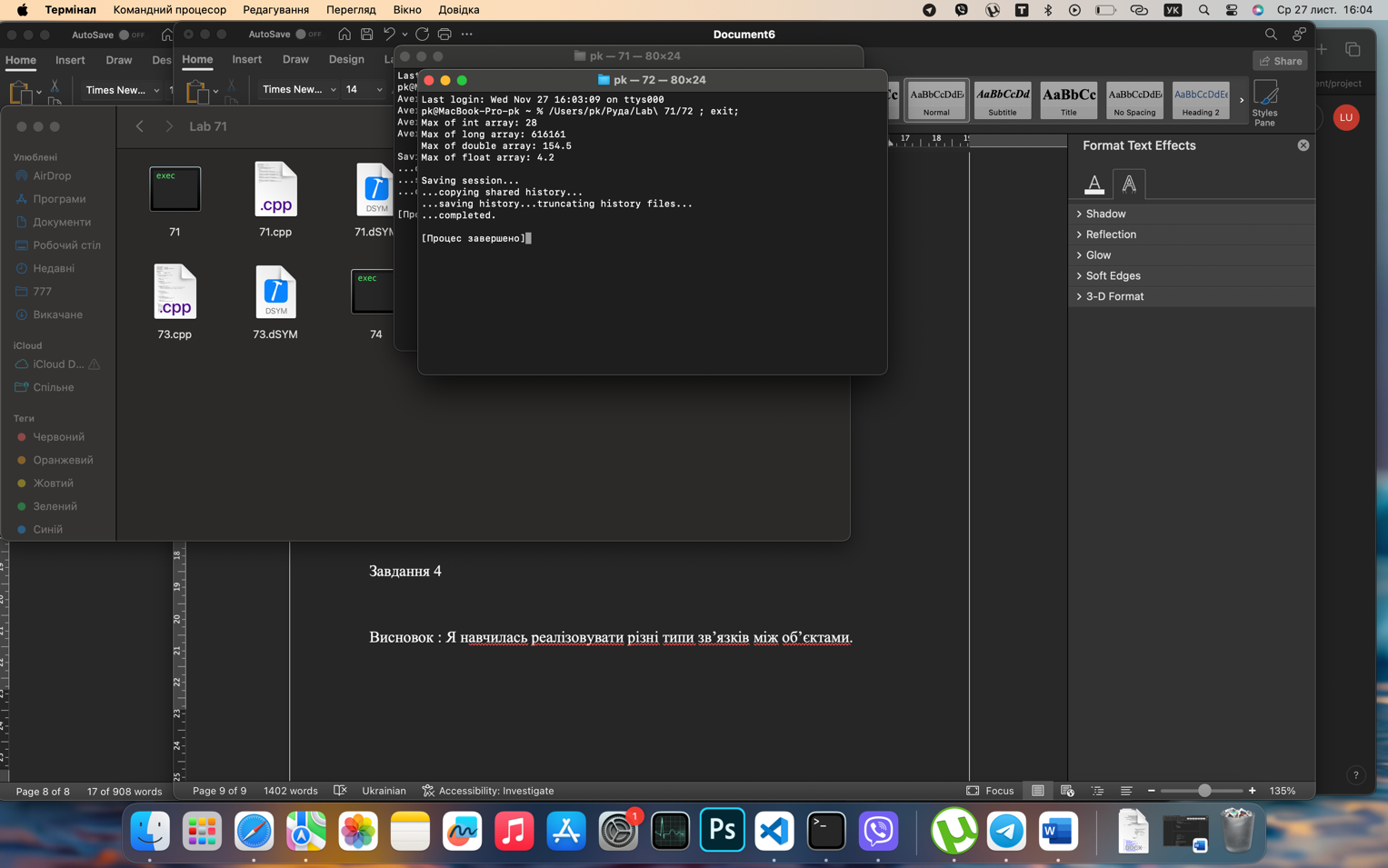
}

Screen-shot вікна виконання програми.

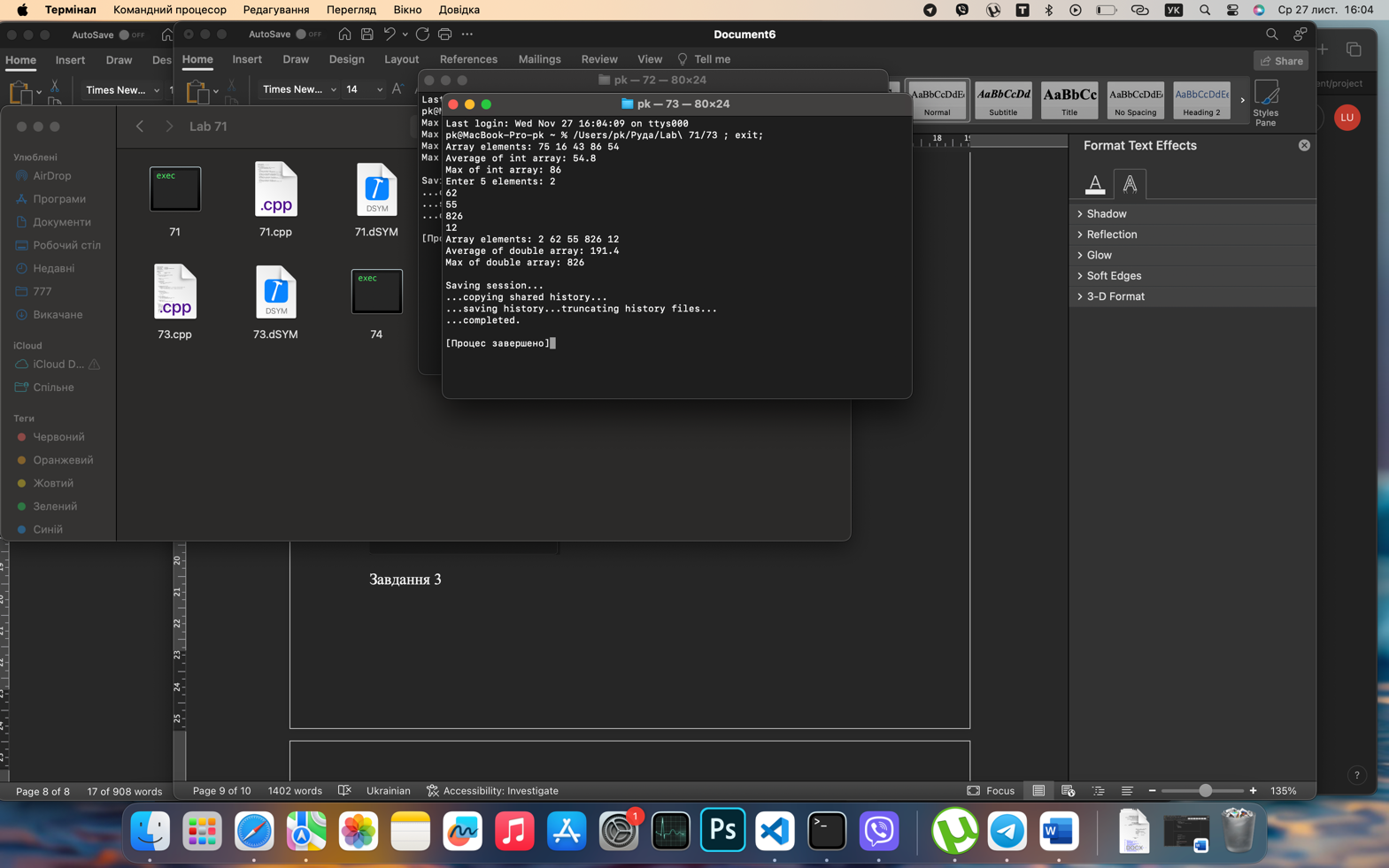
Завдання 1



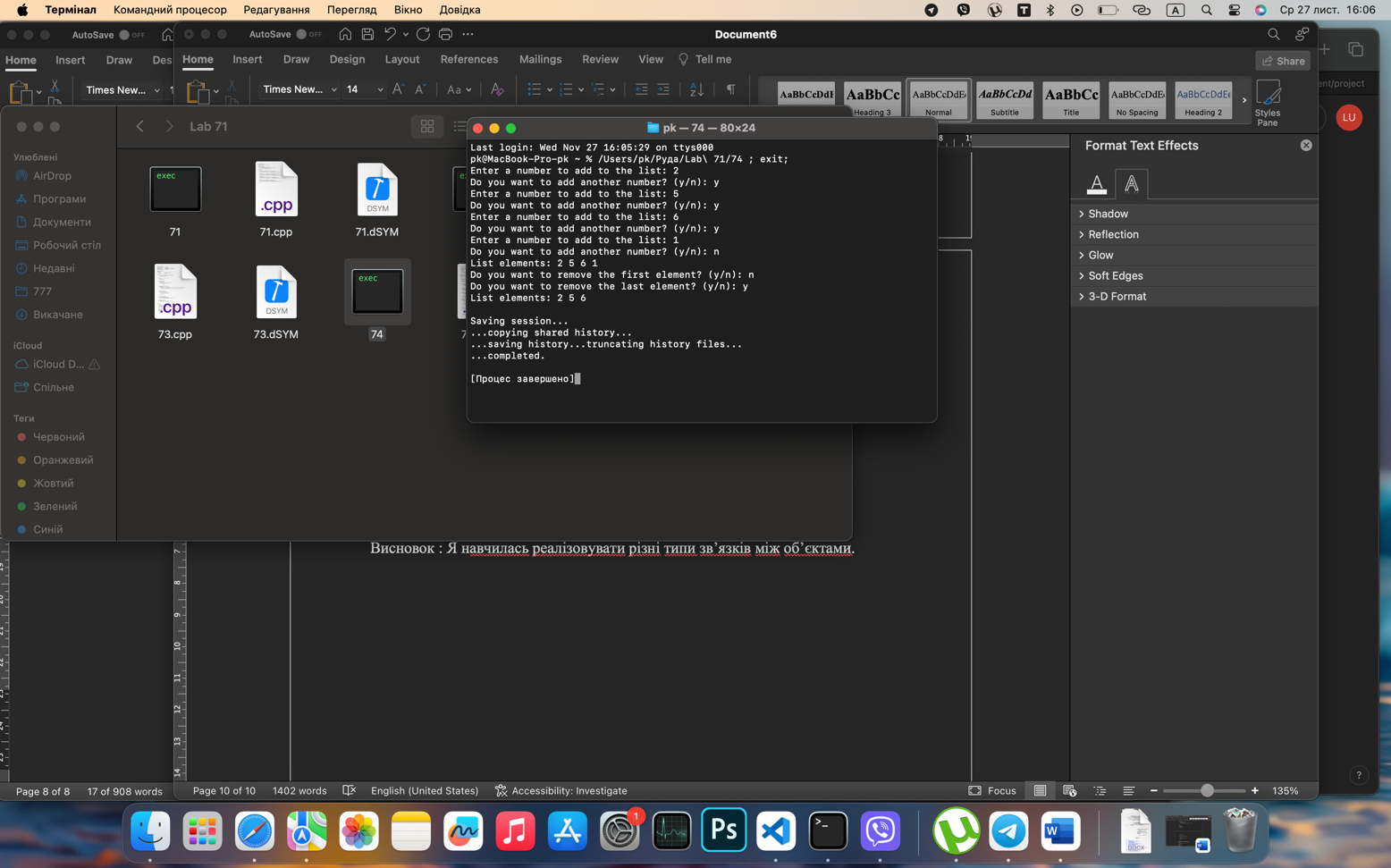
Завдання 2



Завдання 3



Завдання 4



Висновок : Я навчилась створювати і використовувати шаблонні функції і класи