МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет "Львівська політехніка" Кафедра САПР



# Звіт з лабораторної роботи №6

**з дисципліни “Об’єктно-орієнтоване програмування” “Стандартна бібліотека шаблонів\_Smart\_pointer” Варіант - 11**

Виконав: ст. гр. КН-108

Пагута В.О.

Прийняв:

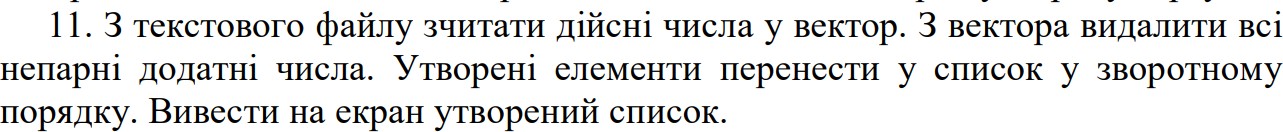
Головатий А.І.

# Львів 2022

**Мета роботи:**

Набути уміння та навички розробки та опису програм з використанням динамічних структур, шаблонів класів та контейнерів. Навчитись застосувати розумні вказівники бібліотеки STL.

# Завдання №1:



**Код:**

#include <iostream> #include <fstream> #include <vector> #include <list> #include <windows.h> using namespace std;

void print(vector<float>& array);

void print(vector<float>& array)

{

if (array.empty())

{

cout << "Вектор порожній";

return;

}

vector<float>::iterator iter;

for (iter = array.begin(); iter != array.end(); iter++)

{

cout << \*iter << " ";

}

}

void print(list<float>& l);

void print(list<float>& l)

{

if (l.empty())

{

cout << "Список порожній";

return;

}

list<float>::iterator iter;

for (iter = l.begin(); iter != l.end(); iter++)

{

cout << \*iter << " ";

}

cout << endl;

}

int main()

{

SetConsoleCP(1251); SetConsoleOutputCP(1251);

string ipath = "file\_input.txt"; ifstream fin;

fin.open(ipath, ios::in); float n;

vector<float> arr;

for (int i = 0; !fin.eof(); i++)

{

fin >> n; arr.push\_back(n);

}

cout << "Зчитані з файлу у вектор елементи: ";

print(arr); cout << endl; fin.close(); int a;

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

{

if(arr[i]>0)

{

a = arr[i];

if(a%2 != 0)

{

arr.erase(arr.begin()+i);

}

}

}

cout << "Видалені непарні додатні числа: ";

print(arr); cout << endl;

list<float> list;

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

{

list.push\_back(arr[i]);

}

cout << "Записано у список: ";

print(list);

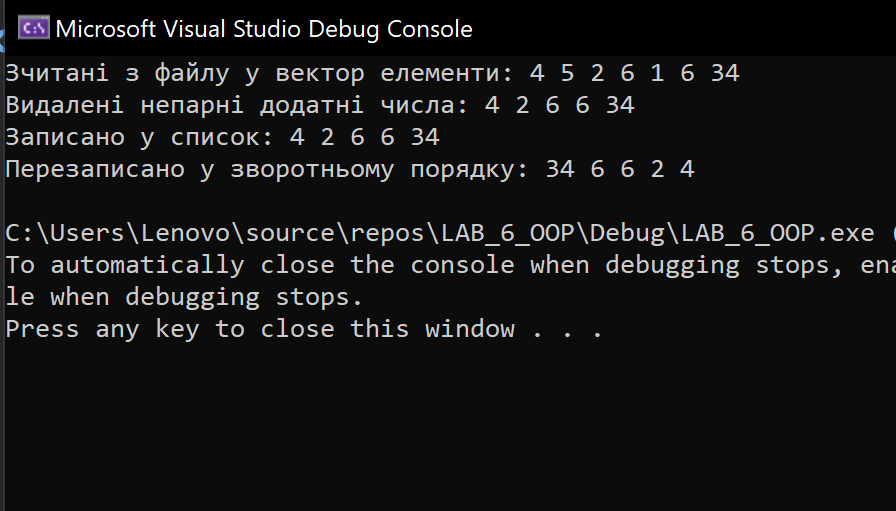
list.reverse();

cout << "Перезаписано у зворотньому порядку: ";

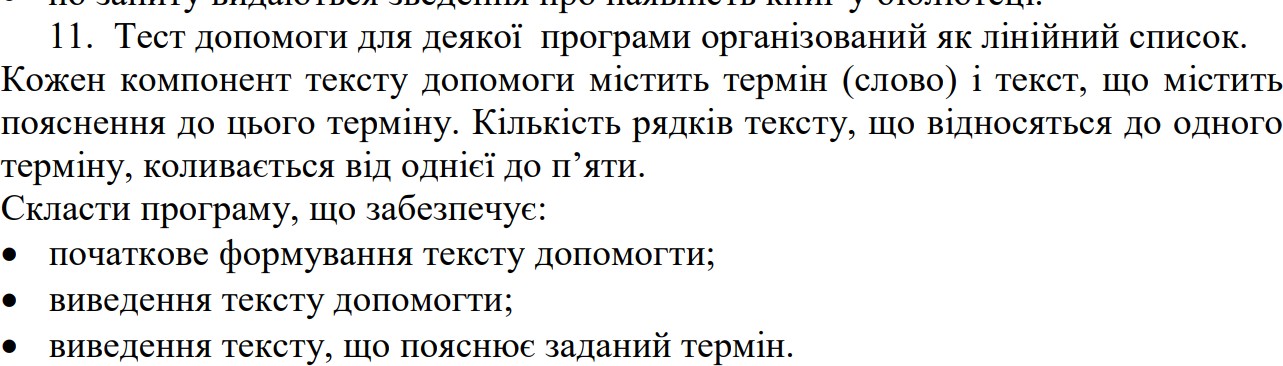
print(list);

}

# Виконання:



**Завдання №2:**



**Код:**

#include <iostream> #include <string> #include <windows.h> #include <list> #include <iterator> using namespace std;

class helpText

{

public:

string term; string meaning;

helpText(string trm,string mean)

{

this->term = trm;

this->meaning = mean;

}

helpText()

{

term = "undefined"; meaning = "undefined";

}

~helpText()

{

}

void printTerm(); void printMeaning();

};

void helpText::printTerm()

{

cout << this->term << " - ";

}

void helpText::printMeaning()

{

cout << this->meaning << "\n";

}

/\*void newElement(list<helpText> lst, string trm, string mean)

{

helpText\* a = new helpText(trm,mean); lst.push\_back(\*a);

}\*/

void print(list<helpText> lst);

void print(list<helpText> lst)

{

if (lst.empty())

{

cout << "Список порожній";

return;

}

for (auto& item : lst) { item.printTerm(); item.printMeaning(); cout << endl;

}

cout << endl;

}

int main()

{

SetConsoleCP(1251); SetConsoleOutputCP(1251); int i = 1;

int choice = 1; list<helpText> lst;

helpText OOP;

OOP.term = "Об'єктно-орієнтовне програмування";

OOP.meaning = "одна з парадигм програмування, яка розглядає програму як множину «об'єктів», що взаємодіють між собою.[4] Основу ООП складають чотири основні концепції: інкапсуляція, успадкування, поліморфізм та абстракція[5] Однією з переваг ООП є краща модульність програмного забезпечення.";

helpText JS;

JS.term = "ДжаваСкріпт";

JS.meaning = "динамічна, об'єктно-орієнтована[4] прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше

використовується для створення сценаріїв вебсторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки.";

helpText Bandera;

Bandera.term = "Степан Андрійович Бандера";

Bandera.meaning = "український політичний діяч, один із чільних

ідеологів і теоретиків українського націоналістичного руху XX століття[1], після розколу Організації українських націоналістів — голова Проводу ОУН-Б[2].";

lst.push\_back(OOP); lst.push\_back(JS); lst.push\_back(Bandera);

while (choice != 0)

{

cout << "Виберіть дію:" << endl; cout << "1.Вибрати термін" << endl;

cout << "2.Вивести весь текст допомоги" << endl;

cout << "3.Створити новий термін" << endl;

cin >> choice; switch (choice)

{

case 1:

cout << "Виберіть термін:" << endl;

for (helpText& item : lst) { cout << i << ". "; item.printTerm();

cout << endl; i++;

}

cin >> choice; i = 1;

for (helpText& item : lst)

{

if (i == choice)

{

item.printTerm(); item.printMeaning();

}

i++;

}

i = 1;

break;

case 2: print(lst); break;

case 3:

string trm; string mean;

cout << "Введіть назву терміну: "; getline(cin, trm); getline(cin, trm); cout << "Введіть значення терміну"; getline(cin, mean);

helpText\* a = new helpText(trm, mean); lst.push\_back(\*a);

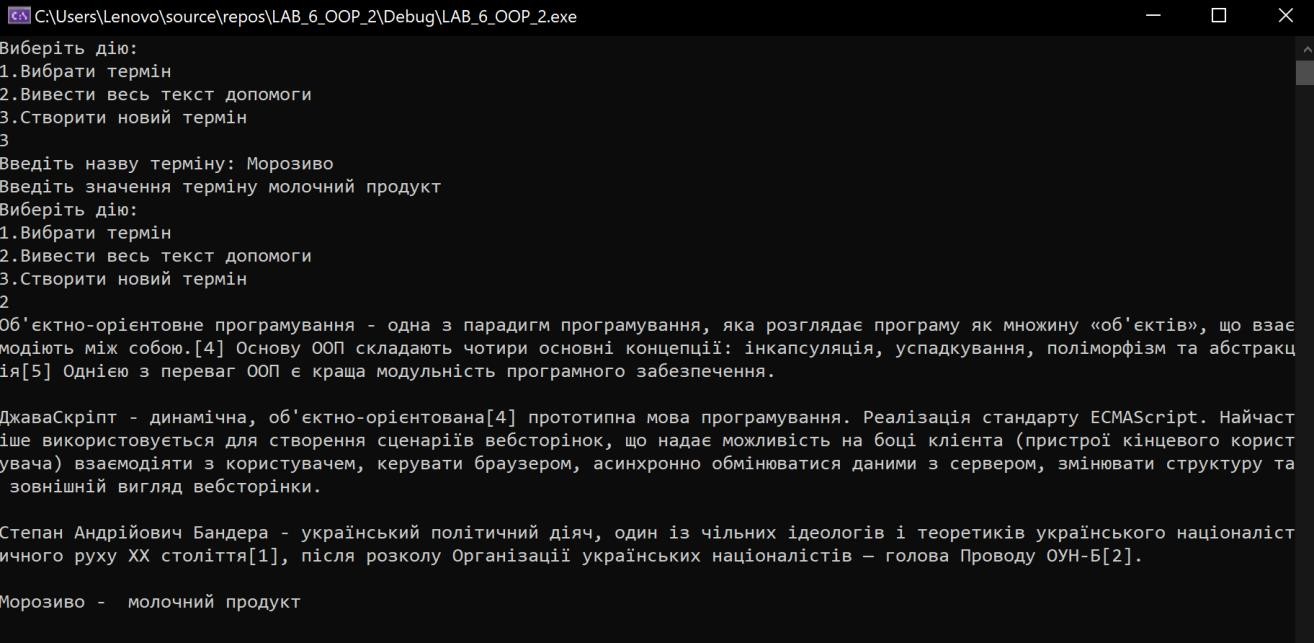
break;

}

}

}

# Виконання:



**Контрольні запитання:**

1. **Що таке динамічні структури? Опишіть їх.**

Динамічні структури за визначенням характеризуються відсутністю фізичної суміжності елементів структури в пам'яті, непостійністю і

непередбачуваністю розміру (кількість елементів) структури в процесі її обробки.

# Як додати новий елемент у список? Приклад.

List<int> lst; Int l = 5;

Lst.push\_back(l);

# Як видалити елемент з черги? Приклад.

1. Lst.pop\_back(l);
2. Lst.pop\_front(l);

# Як додати елемент у стек? Приклад.

stack<int> stck; Int l = 3; Stck.push(l);

# Що таке дек? Приклад.

Дек - двухсторонній список.

deque<**char**> myDeque(5);

1. **Як включити новий елемент у бінарне дерево? Приклад.**

node\* root = NULL; int p;

int key;

cout << "Enter amount of keys: "; cin >> p;

while (p != 0)

{

cin >> key;

root = insert(root, key); p--;

}

1. **Що таке контейнер? Види та їх призначення.**

Контейнери — це об’єкти, що зберігають інші елементи і реалізують механізми доступу до них. Прикладами контейнерів є вектори і списки. Кожний контейнер описується

шаблонним класом, у якому реалізуються механізми доступу і функція для обробки елементів, що містяться у контейнері. Кожний контейнер має свої ітератори для

перебору елементі і функції для їх обробки. Наприклад, клас, що описує роботу із вектором, має відповідний ітератор для прямого доступу до елементів вектору, а також функції для вставки і видалення тощо.

1. **Для чого призначений вектор? Які методи він підтримує?**

Вектор — це абстрактна модель, яка імітує динамічний масив.

|  |  |
| --- | --- |
| vector::clear | Видаляє всі елементи  вектора |
| vector::insert | Вставка елементів у вектор |
| vector::erase | Видалення вказаних  елементів вектора (один або  декілька) |
| vector::push\_back | Вставка елемента в кінець  вектора |
| vector::pop\_back | Видалити останній елемент  вектора |
| vector::resize | Змінює розмір вектора на  задану величину |
| vector::swap | Обміняти вміст двох векторів |

1. **Наведіть два випадки застосування typename.?**

Для ініціалізації шаблонної функції.

1. **Як використовується слово export стосовно шаблонів?**

Ключове слово export може передувати оголошенню template. Воно дозволяє

використовувати шаблон з іншого файлу, повторюючи лише його оголошення, а не усе визначення.

**11,12.Що таке розумний вказівник?**

Це абстрактний тип даних, який імітує вказівник з допоміжними можливостями, такими як автоматичне керування пам'яттю або перевірку виходу за межі виділеної пам'яті.

# Висновок:

Я набув уміння та навички розробки та опису програм з використанням динамічних структур, шаблонів класів та контейнерів, а також навчився застосовувати розумні вказівники бібліотеки STL.