МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

ЗВІТ

До лабораторної роботи №2
На тему: *"Документування етапів проектування та кодування програми"*
З дисципліни: *"Вступ до інженерії програмного забезпечення"*

Лекторка:
доцент каф. ПЗ
Левус Є.В.

Виконав:
ст. гр. ПЗ-11
Солтисюк Д.А.

Прийняла:
доцент каф. ПЗ
Левус Є.В.

«____» _____ 2022р.

∑ =

Львів – 2022

Тема. Документування етапів проектування та кодування програми.
Мета. Навчитися документувати основні результати етапів проектування та кодування найпростіших програм.

Теоретичні відомості

Варіант 24

28. Як записуються класи та їх складові у мові С++?
Спираючись на загальноприйняті правила оформлення зрозумілого, читабельного коду на мові програмування С++, можна сказати, що класи у тексті програми мають бути записані у стилі “PascalCase”. Самі назви складових класу повинні дотримуватись стилю “camelCase”.
Приклад записування класу:

```
class Person {

public:

string name;

time_t bornAt;

}
```

16. Скільки входів та виходів має блок циклу?
Оператор циклу в блок системах позначається шестикутником. Він містить дві точки входу та одну точку виходу. Першою є, безпосередньо, точка входу в цикл при наближенні процесу виконання до цього оператора, другою є точка входу при ітераційному процесі. Вихід з циклу можливий двома способами - при справдженні умови виходу та при використанні оператора *break.* Проте ці два способи поділяють одну точку виходу.

10. Що таке алгоритм?
Алгоритм – набір інструкцій, які описують порядок дій виконавця, щоб досягти результату розв'язання задачі за скінченну кількість дій. Якщо спрощено: алгоритм задає систему правил, за якою вхідна інформація перетворюються у вихідну. Прикладами алгоритму можуть слугувати інструкції користування смартфоном, надання першої медичної допомоги, приготування страв, виконання арифметичних дій або гімнастичних вправ, покроковий план розв'язування задачі або проведення фізичного чи хімічного досліду.

Постановка завдання

Частина І. У розробленій раніше програмі до лабораторної роботи з дисципліни «Основи програмування» внести зміни – привести її до модульної структури, де модуль – окрема функція-підпрограма. У якості таких функцій

запрограмувати алгоритми зчитування та запису у файл, сортування, пошуку, редагування, видалення елементів та решта функцій згідно варіанту.
Частина ІІ. Сформувати пакет документів до розробленої раніше власної програми:

1. схематичне зображення структур даних, які використовуються для збереження інформації ;
2. блок-схема алгоритмів – основної функції й двох окремих функцій-підпрограм (наприклад, сортування та редагування);
3. текст програми з коментарями та оформлений згідно наведених рекомендацій щодо забезпечення читабельності й зрозумілості.

Для схематичного зображення структур даних, блок-схеми алгоритму використати редактор MS-Visio або інший редактор інженерної та ділової графіки.

Отримані результати

1.Схематичне зображення структур даних, які використовуються для збереження інформації:

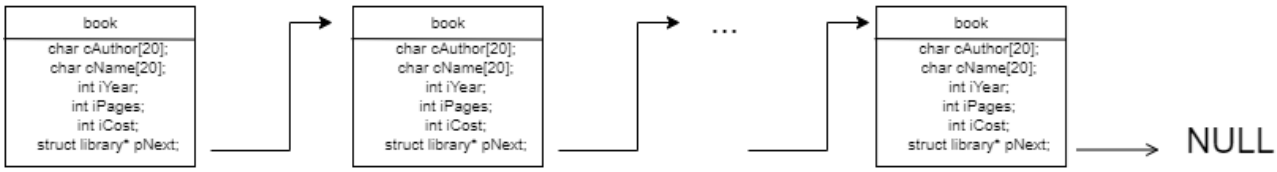

Рис.1. Графічне зображення однозв’язного списку структур book

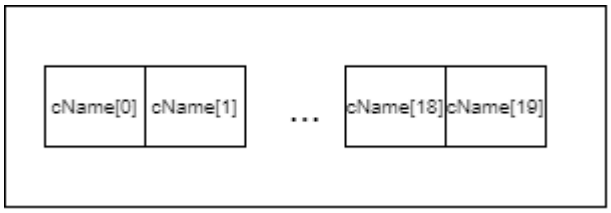

Рис.2. Графічне зображення одновимірного масиву

2. Блок-схема алгоритмів:

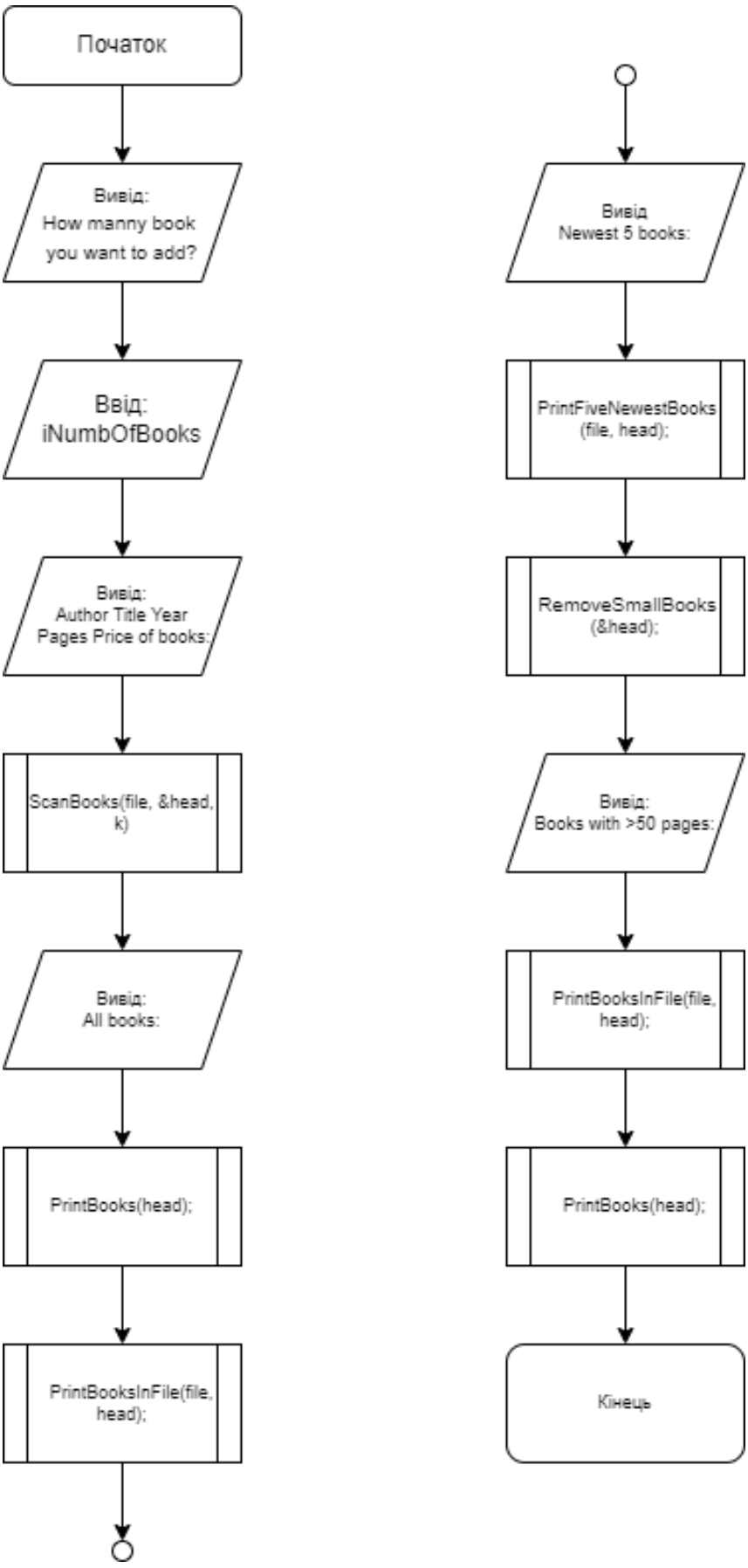

Рис.3. Блок-схема функції main

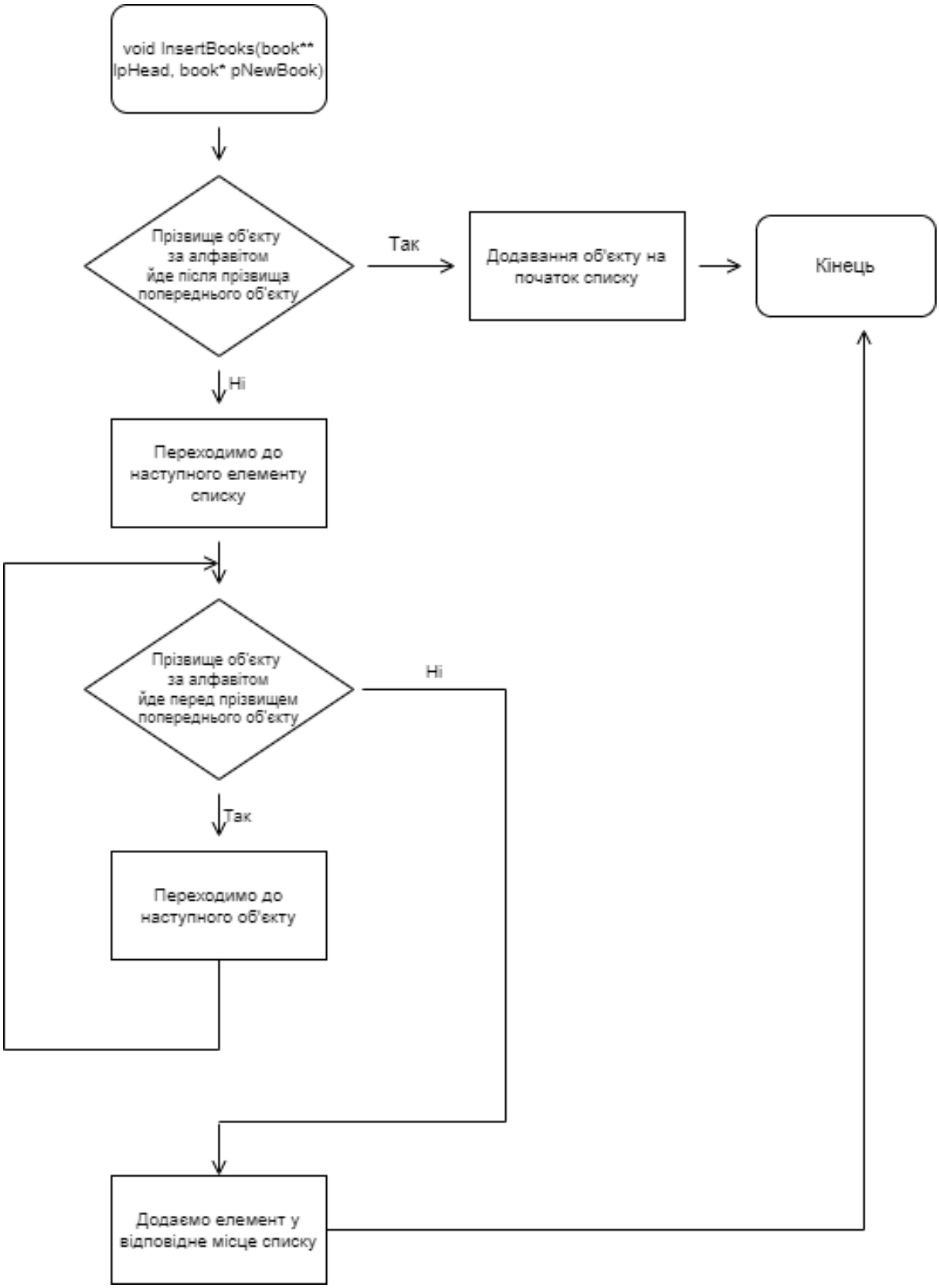

Рис.4. Блок-схема функції сортування

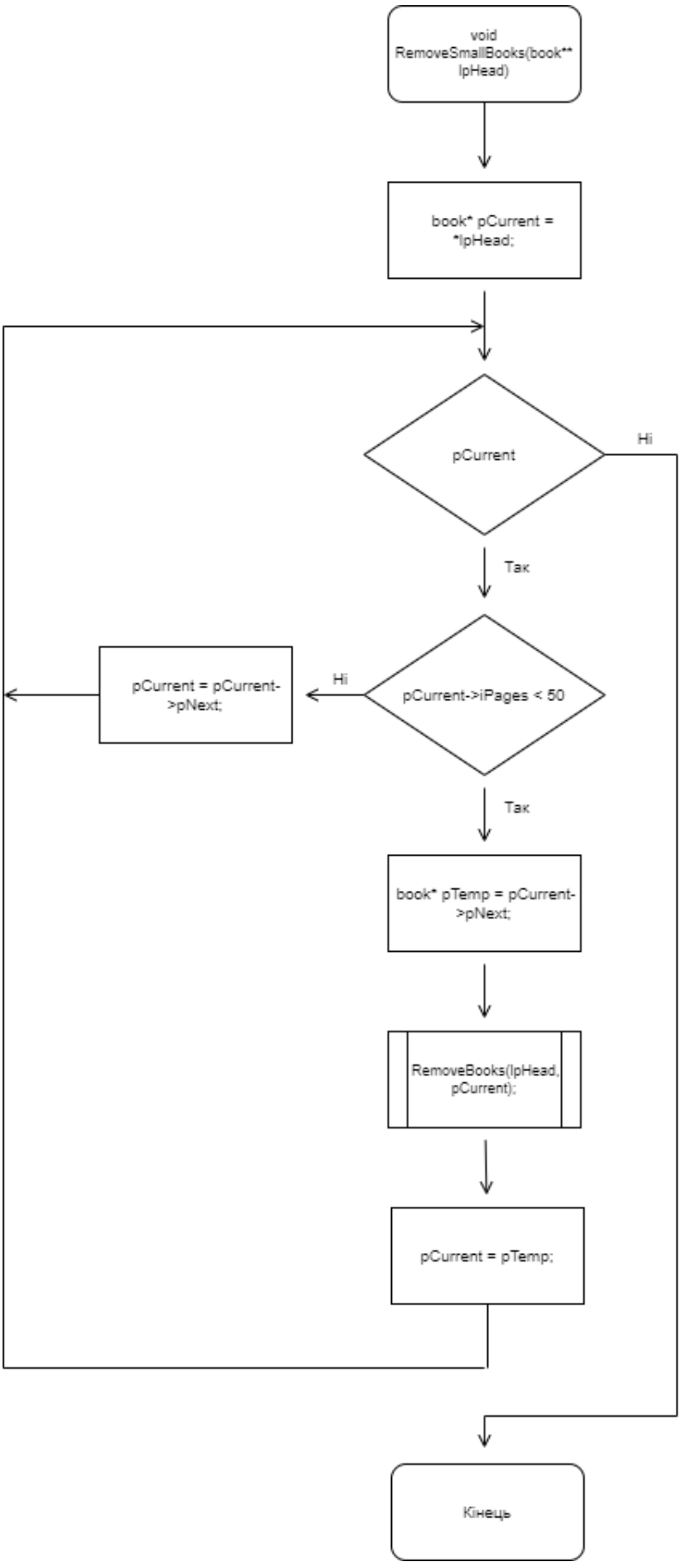

Рис.5. Функція видалення книг, що мають менше 50-ти сторінок

Текст програми з коментарями:

```
#define _CRT_SECURE_NO_WARNINGS
#define N 8
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

// Структура для збереження даних про книгу
// Приклад використання:
//  book* pNode;
//  scanf("%s %s %d %d %d", pNode->CAuthor, pNode->cName,
//      &pNode->iYear, &pNode->iPages, &pNode->iCost);
typedef struct library
{
    char CAuthor[20];
    char cName[20];
    int iYear;
    int iPages;
    int iCost;
    struct library *pNext;
}book;


void PrintBooks(book*);
void PrintBooksInFile(FILE* file, book* pHead);
int ScanBooks(book** lpHead, int iNumbOfBooks);
int ScanBooksFromFile(FILE* file, book** lpHead);
void InsertBooks(book** lpHead, book* pNewBook);
void RemoveBooks(book** lpHead, book* pNode);
void RemoveSmallBooks(book** lpHead);
void FreeLibrary(book** lpHead);
void PrintFiveNewestBooks(FILE* file, book* pHead);

int main()
{
    FILE* file;
    file = fopen("file.txt", "r");
    book* pHead = NULL;
    int iNumbOfBooks = 0;

    if (file == NULL)
        return 0;
    if (ScanBooksFromFile(file, &pHead))
        return 0;
    fclose(file);

    printf("How manny book you want to add?\n");
    if (!scanf("%i", &iNumbOfBooks))
        return 0;
    if (iNumbOfBooks > 0)
        printf("Enter \"Author Title Year Pages Price\" of %i books:\n",
iNumbOfBooks);
    if (ScanBooks(&pHead, iNumbOfBooks))
        return 0;

    printf("All books:\n");
```

```
    PrintBooks(pHead);

    file = fopen("output.txt", "w");
    if (file == NULL)
        return 0;
    fprintf(file, "All books:\n");
    PrintBooksInFile(file, pHead);

    printf("Newest 5 books:\n");
    fprintf(file, "Newest 5 books:\n");
    PrintFiveNewestBooks(file, pHead);

    RemoveSmallBooks(&pHead);

    printf("Books with >50 pages:\n");
    fprintf(file, "Books with >50 pages:\n");
    PrintBooksInFile(file, pHead);
    PrintBooks(pHead);

    fclose(file);

    FreeLibrary(&pHead);

    fclose(file);
}
//---------------------------------------------------------------------------
// Вивід в консоль всіх записаних книг з файлу
void PrintBooks(book* pHead) {
    book* pCurrent = pHead;
    while (pCurrent) {
        printf("%s %s %d %d %d\n", pCurrent->cAuthor, pCurrent->cName,
            pCurrent->iYear, pCurrent->iPages, pCurrent->iCost);
        pCurrent = pCurrent->pNext;
    }
    printf("\n");
}
//---------------------------------------------------------------------------
// Запис усіх книг (компонентів однозв'язному списку) у файл
void PrintBooksInFile(FILE* file, book* pHead) {
    book* pCurrent = pHead;
    while (pCurrent) {
        fprintf(file, "%s %s %d %d %d\n", pCurrent->cAuthor, pCurrent->cName,
            pCurrent->iYear, pCurrent->iPages, pCurrent->iCost);
        pCurrent = pCurrent->pNext;
    }
    fprintf(file, "\n");
}
//---------------------------------------------------------------------------
// Зчитування книг з файлу
int ScanBooksFromFile(FILE* file, book** lpHead) {
    book* pNode;
    for (int i = 0; i < N; i++) {
        pNode = (book*)malloc(sizeof(book));
        if (!pNode) {
            return 1;
        }
        pNode->pNext = NULL;
        if (!fscanf(file, "%s %s %d %d %d", pNode->cAuthor, pNode->cName,
            &pNode->iYear, &pNode->iPages, &pNode->iCost))
            return 1;
```

```
        InsertBooks(lpHead, pNode);
    }
    return 0;
}
//-------------------------------------------------------------------------
// Зчитування книг з консолі
int ScanBooks(book** lpHead, int iNumbOfBooks) {
    book* pNode;
    for (int i = 0; i < iNumbOfBooks; i++) {
        pNode = (book*)malloc(sizeof(book));
        if (!pNode) {
            return 1;
        }
        pNode->pNext = NULL;
        if (!scanf("%s %s %d %d %d", pNode->CAuthor, pNode->cName,
            &pNode->iYear, &pNode->iPages, &pNode->iCost))
            return 1;
        InsertBooks(lpHead, pNode);
    }
    return 0;
}
//-------------------------------------------------------------------------
// Повертає посортований список книг по прізвищу в протилежному до алфавітного
порядку
void InsertBooks(book** lpHead, book* pNewBook)
{
    if (!*lpHead || strcmp((*lpHead)->CAuthor, pNewBook->CAuthor) < 0) {
        pNewBook->pNext = *lpHead;
        *lpHead = pNewBook;
        return;
    }

    book* pCurrent = *lpHead;
    while (pCurrent->pNext && strcmp(pCurrent->pNext->CAuthor, pNewBook-
>CAuthor) > 0)
    {
        pCurrent = pCurrent->pNext;
    }
    pNewBook->pNext = pCurrent->pNext;
    pCurrent->pNext = pNewBook;
}
//-------------------------------------------------------------------------
// Видалення книг (компонентів однозв'язного списку)
void RemoveBooks(book** lpHead, book* pNode)
{
    if (!*lpHead || !pNode)
        return;
    book* pTemp = NULL;

    if (*lpHead == pNode) {
        pTemp = *lpHead;
        *lpHead = (*lpHead)->pNext;
        free(pTemp);
        return;
    }

    book* pCurrent = *lpHead;
    while (pCurrent) {
        if (pCurrent->pNext == pNode) {
            pTemp = pCurrent->pNext;
```

```
            pCurrent->pNext = pCurrent->pNext->pNext;
            free(pTemp);
            return;
        }
        pCurrent = pCurrent->pNext;
    }
}
//-------------------------------------------------------------------------
// Видалення книг, які мають менше 50-ти сторінок
void RemoveSmallBooks(book** lpHead)
{
    book* pCurrent = *lpHead;

    while (pCurrent)
    {
        if (pCurrent->iPages < 50) {
            book* pTemp = pCurrent->pNext;
            RemoveBooks(lpHead, pCurrent);
            pCurrent = pTemp;
            continue;
        }
        pCurrent = pCurrent->pNext;
    }
}
//-------------------------------------------------------------------------
// Очищення структури
void FreeLibrary(book** lpHead)
{
    book* pCurrent = *lpHead;
    while (pCurrent)
    {
        book* pTemp = pCurrent;
        pCurrent = pCurrent->pNext;
        free(pTemp);
    }
}
//-------------------------------------------------------------------------
// Виведення в консоль 5-ти найновіших книг за роком видання
void PrintFiveNewestBooks(FILE* file, book* pHead)
{
    book* pCurrent = pHead;
    int max = pCurrent->iYear, prev_max;
    while (pCurrent) {

        if (pCurrent->iYear > max)
            max = pCurrent->iYear;
        pCurrent = pCurrent->pNext;
    }
    for (int i = 0; i < 5;) {
        while (max > 0) {
            pCurrent = pHead;

            while (pCurrent) {
                if (pCurrent->iYear == max) {
                    printf("%s %s %d %d %d\n", pCurrent->CAuthor, pCurrent-
>cName,
                        pCurrent->iYear, pCurrent->iPages, pCurrent->iCost);
                    fprintf(file, "%s %s %d %d %d\n", pCurrent->CAuthor,
pCurrent->cName,
                        pCurrent->iYear, pCurrent->iPages, pCurrent->iCost);
```

```
                    i++;
                    max--;
                    break;
                }
                pCurrent = pCurrent->pNext;
            }
            max--;
            break;
        }
    }
    printf("\n");
    fprintf(file, "\n");
}
```

Висновок

На даній лабораторній роботі я переписав код з використанням угорської нотації. Також, я навчився документувати основні результати етапів проектування та кодування. Вивчені знання можу використати в подальшому проектуванні архітектури ПЗ.