

Національний університет харчових технологій

(повне найменування вищого навчального закладу)

Інформаційних систем

(повна назва кафедри, циклової комісії)

КУРСОВА РОБОТА

з дисципліни Основи програмування та алгоритмічні мови

(назва дисципліни)

на тему: Створення консольної інформаційної системи: Довідник
гелікоптерів

Студента (ки) 1 курсу 2 групи
спеціальності 122 "Комп'ютерні науки "
Держій Д.Ю

Керівник Грибков С.В.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____ Оцінка: ECTS _____

Члени комісії

_____	_____
(підпис)	(прізвище та ініціали)
_____	_____
(підпис)	(прізвище та ініціали)
_____	_____
(підпис)	(прізвище та ініціали)

Я, як здобувач(-ка) Національного університету харчових технологій, розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

(підпис)

Київ-2023

(назва вищого навчального закладу)

Кафедра Інформаційних систем

Дисципліна Основи програмування та розробки алгоритмів

Спеціальність 122 «Комп'ютерні науки»

Курс 1 Група 2 Семестр II

ЗАВДАННЯ

на курсову роботу студента

Д е р ж і й Д е н и с Ю р і й о в и ч

(прізвище, ім'я, по-батькові)

1. Тема роботи Створення консольної інформаційної системи обліку довідника гелікоптерів

2. Термін здачі студентом закінченого роботи
08.06.2023

3. Вихідні дані до роботи 1. Структура запису: модель, виробник, тип, кількість пасажирських місць, вантажопідйомність, рік випуску. 2.Перелік довідок для пошуку запитів

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

Розділ 1 «Постанова задачі»

Розділ 2 «Опис алгоритмів та програм»

Розділ 3 «Блок-схеми»

Розділ 4 «Текст програми»

Розділ 5 «Інструкція користувача»

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

1. Блок-схеми алгоритмів

2. Скріншоти інтерфейсу користувача

6. Дата видачі завдання

1.05.2023

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів курсової роботи	Термін виконання етапів курсової роботи	Примітка
1.	Аналіз поставленої задачі		
2.	Вибір методів вирішення задачі, опис структур та алгоритмів		
3.	Створення консольної інформаційної системи		
4.	Оформлення пояснювальної записки		
5.	Захист курсової роботи		

Студент _____ Держій. Д. Ю
 (підпис) (прізвище та ініціали)

Керівник курсової роботи _____
 (підпис) (прізвище та ініціали)

АНОТАЦІЯ

Держій Д.Ю. Створення консольної інформаційної системи обліку гелікоптерів.

Курсова робота складається з 39 сторінок, 1 таблиці, 13 рисунків, 9 блок-схем та 10 літературних джерел.

В даній курсовій роботі створена консольна інформаційна система обліку гелікоптерів, що забезпечує отримання цілої низки різної довідкової інформації про виробників, типи гелікоптерів. Під час виконання курсової роботи проведено аналіз предметної області по обліку надходження серіалів за допомогою інтернету, розроблено блок-схеми алгоритмів підпрограм для реалізації функцій системи, створено текстовий інтерфейс користувача.

Інформаційна система створена у середовищі Visual Studio та з використанням мови C\C++.

Ключові слова: ГЕЛІКОПТЕРИ, ІНФОРМАЦІЙНА СИСТЕМА, ПІДПРОГРАМИ, ПРОГРАМНІ БІБЛІОТЕКИ.

SUMMARY

Derzhii D.Y. Creation of a consolidated information system for accounting of helicopters.

The term paper consists of 39 pages, 1 table, 13 figures, 9 flowcharts and 10 references.

In this course work, a console information system for helicopter accounting was created, which provides a number of different reference information about manufacturers and types of helicopters. In the course of the course work, the subject area of accounting for the receipt of serials via the Internet was analysed, flowcharts of subroutine algorithms for implementing the system functions were developed, and a textual user interface was created.

The information system was created in the Visual Studio environment and using the C\C++ language.

Keywords: HELICOPTERS, INFORMATION SYSTEM, SUBROUTINES, SOFTWARE LIBRARIES.

Розділ №1. Постановка задачі

Для створення консольної інформаційної системи, яка задовольняє вимогам, потрібно використовувати мову програмування C/C++.

Забезпечити функції додавання, редагування, видалення, пошуку даних за заданими параметрами, а також реалізувати текстовий інтерфейс користувача, в якому реалізовані дані функції. При виборі пункту меню "Вихід", забезпечити збереження всіх змін і закриття відкритих файлів.

Розділ №2. Опис алгоритмів і програм

Основна частина всього коду є простою і не потребує докладного аналізу. Програма переглядає кожен запис у файлі і, якщо він відповідає вимогам параметрів, виводить його. Дані про гелікоптери повинні містити інформацію про їхню вантажопідйомність, тип, кількість пасажирських місць, виробника та рік випуску.

void tsk1(): Процедура, що забезпечує виведення списку гелікоптерів із заданими кількістю пасажирських місць і типу: Користувач повинен ввести кількість пасажирських місць і тип гелікоптера, і програма повинна вивести список гелікоптерів, які мають введену кількість пасажирських місць та тип.

void tsk2(): Процедура, що забезпечує виведення списку гелікоптерів заданого типу: Користувач повинен ввести тип гелікоптера, і програма повинна вивести список гелікоптерів, які відповідають введеному типу.

void tsk3(): Процедура, що забезпечує виведення списку гелікоптерів із заданою вантажопідйомністю: Користувач повинен мати можливість ввести вантажопідйомність, і програма повинна вивести список гелікоптерів, які мають вантажопідйомність, що задовольняє введеному значенню.

void tsk4(): Процедура, що забезпечує виведення списку гелікоптерів заданого виробника: Користувач повинен ввести назву виробника гелікоптера, і програма повинна вивести список гелікоптерів, які були випущені цим виробником.

void tsk5(): Процедура, що забезпечує виведення списку гелікоптерів, що випустили до/після заданого року: Користувач повинен ввести рік, і програма повинна вивести список гелікоптерів, які були випущені до і після введеного року.

void tsk6(): Процедура, що забезпечує виведення списку гелікоптерів, які випустили у заданий період: Користувач повинен ввести початковий та кінцевий рік, і програма повинна вивести список гелікоптерів, які були випущені протягом цього періоду.

Розділ №3. Блок-схеми

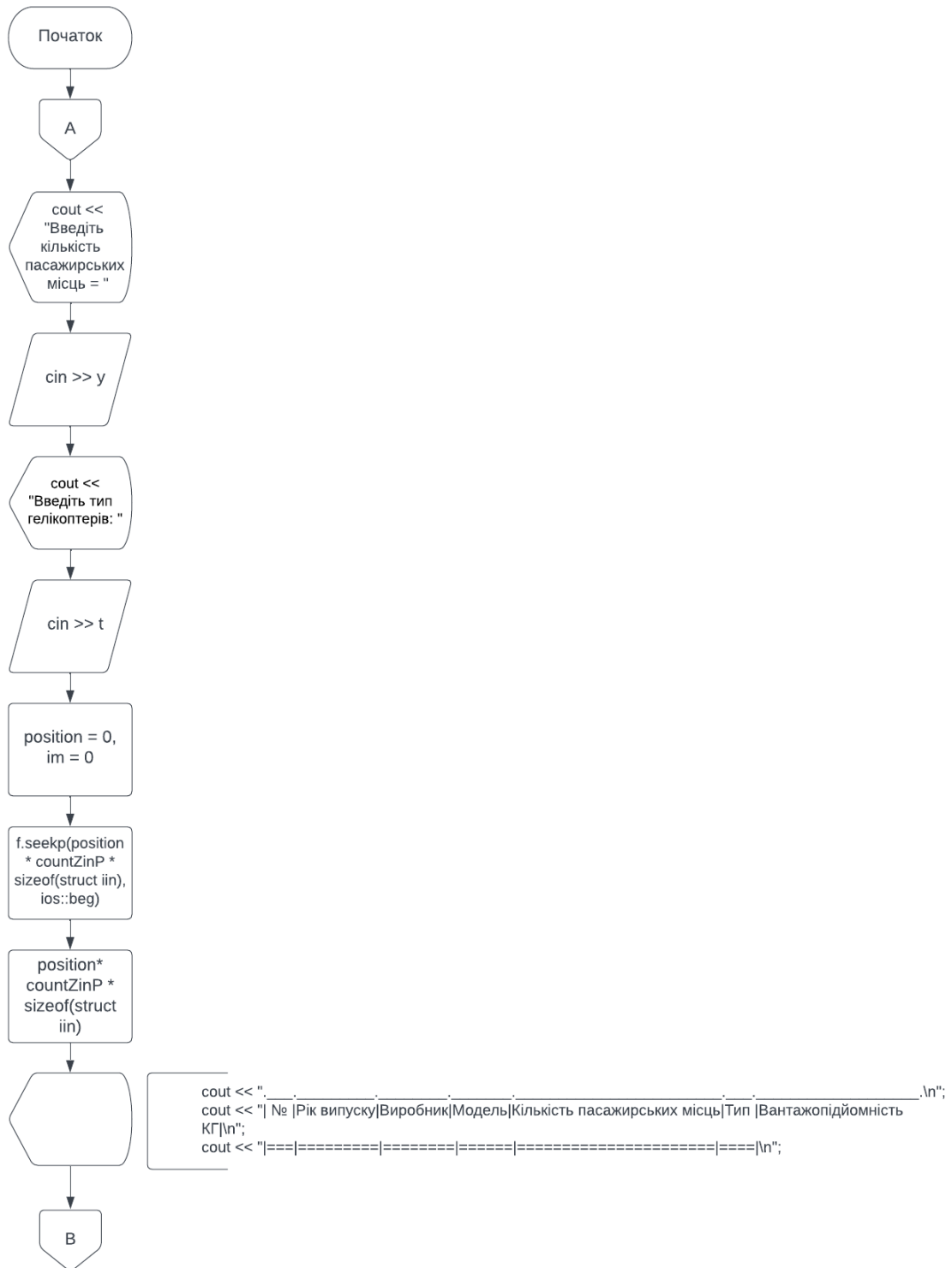


Рис 3.1. Блок-схема до процедури tsk1()

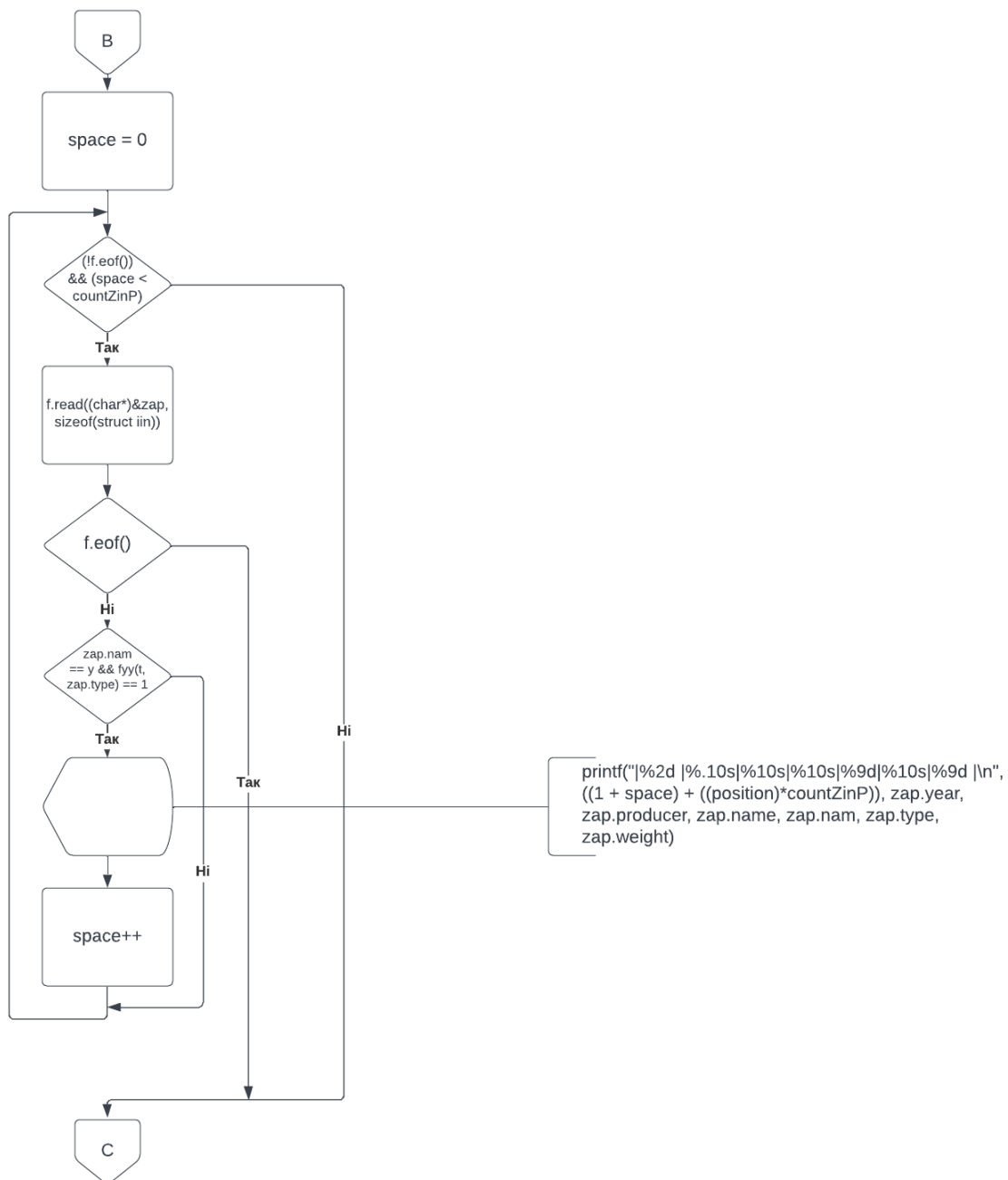


Рис 3.2. Блок-схема до процедури tsk1()

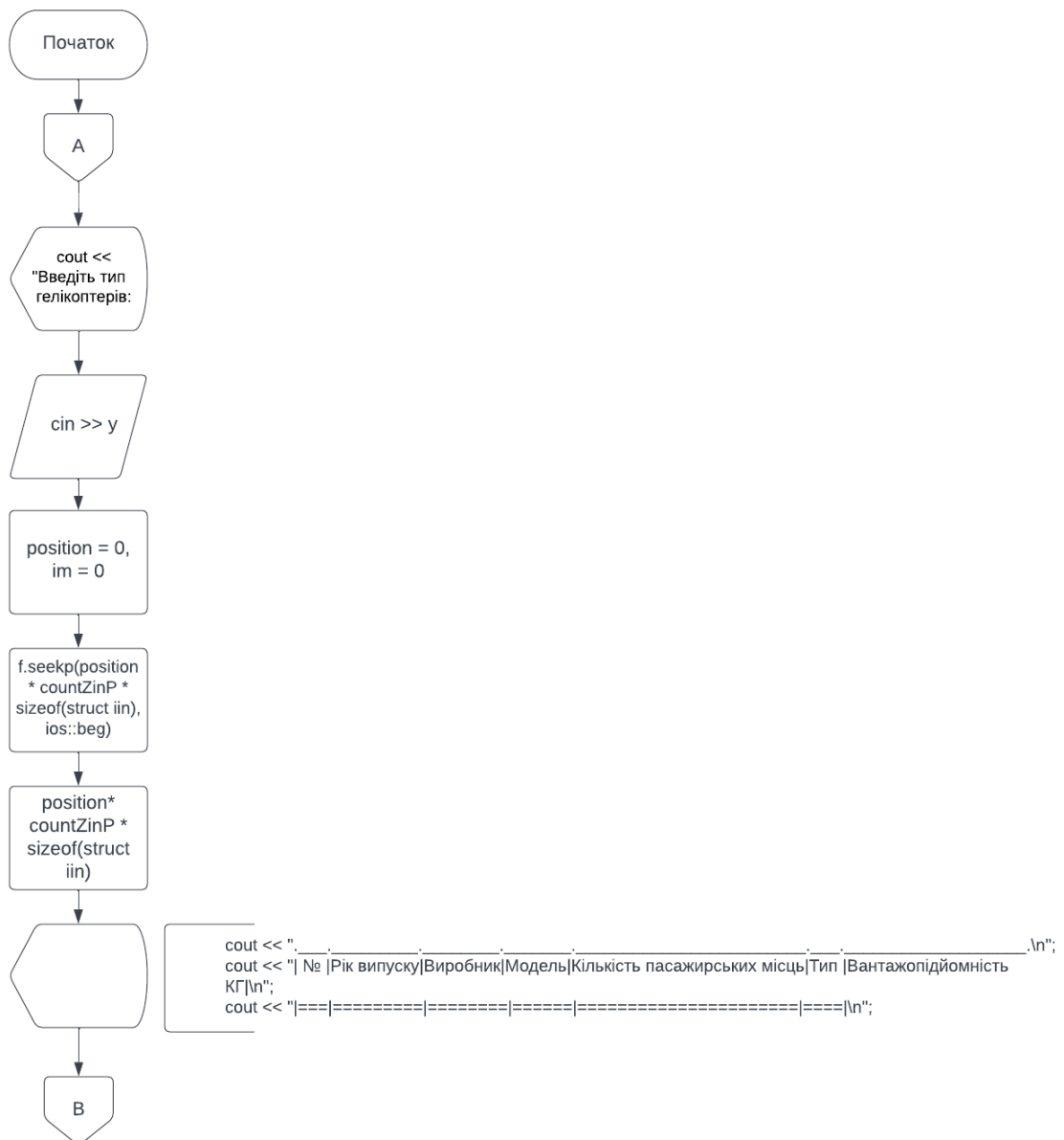


Рис 3.4. Блок-схема до процедури tsk2()

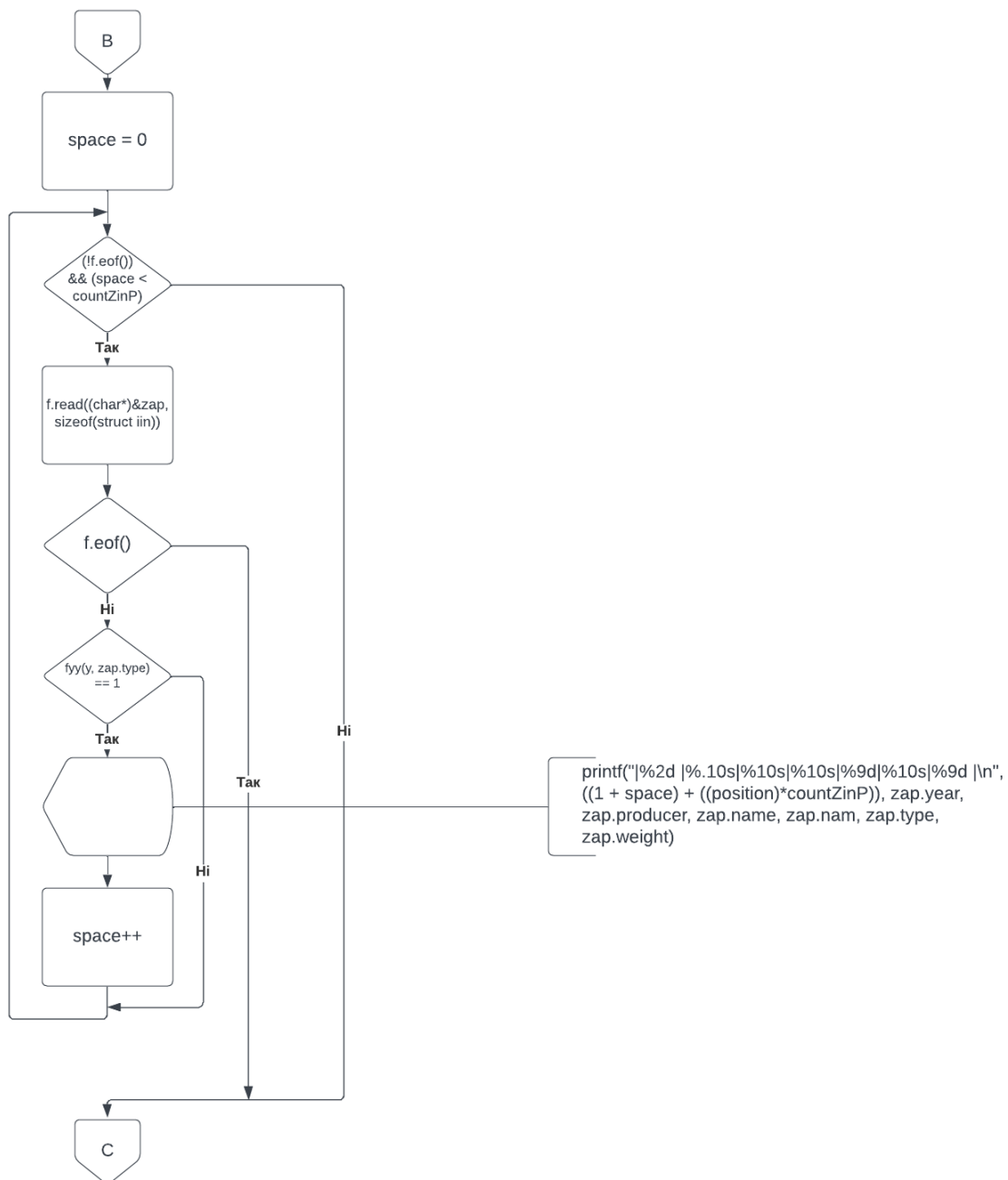


Рис 3.5. Блок-схема до процедури tsk2()

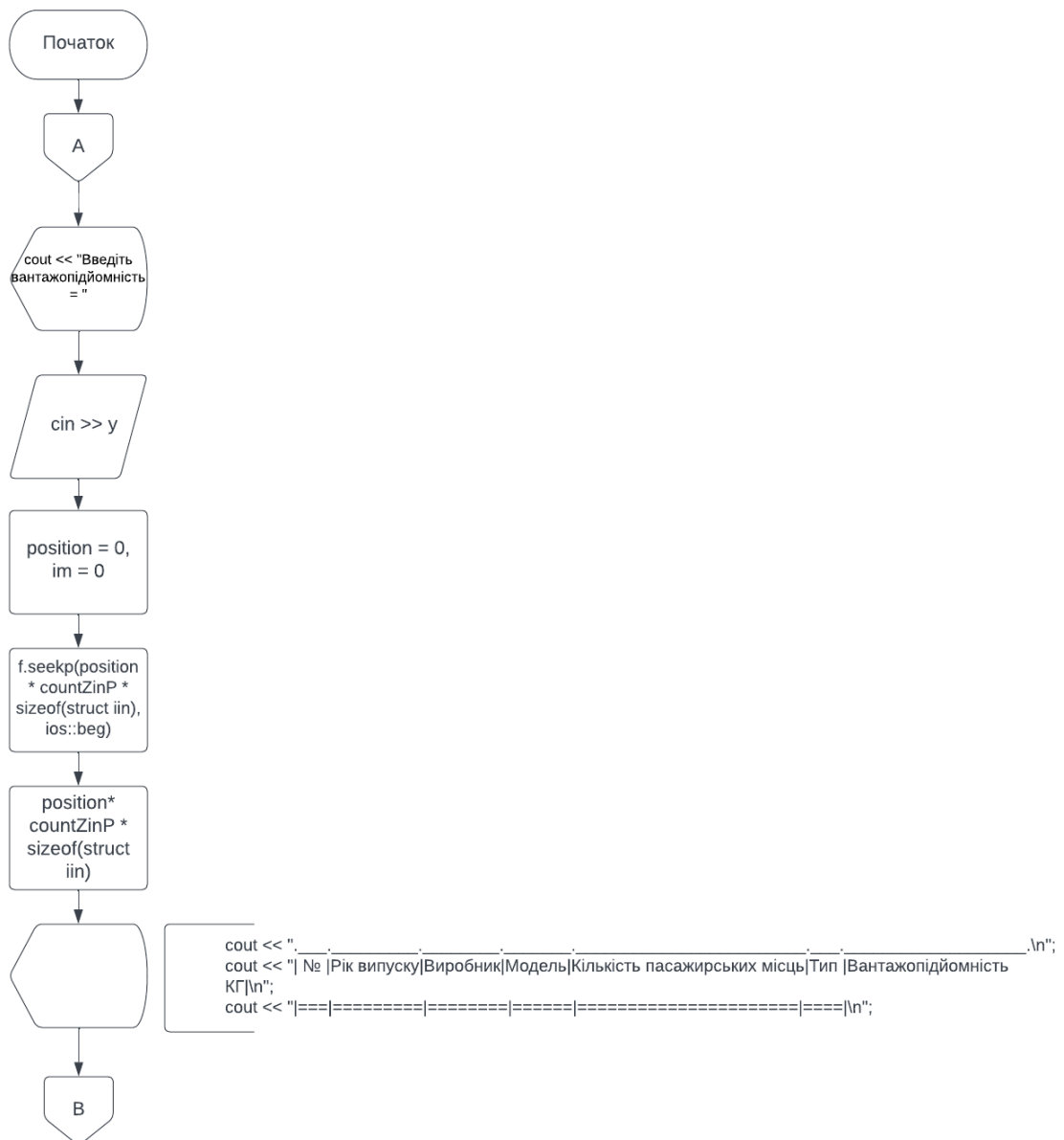


Рис 3.7. Блок-схема до процедури tsk3()

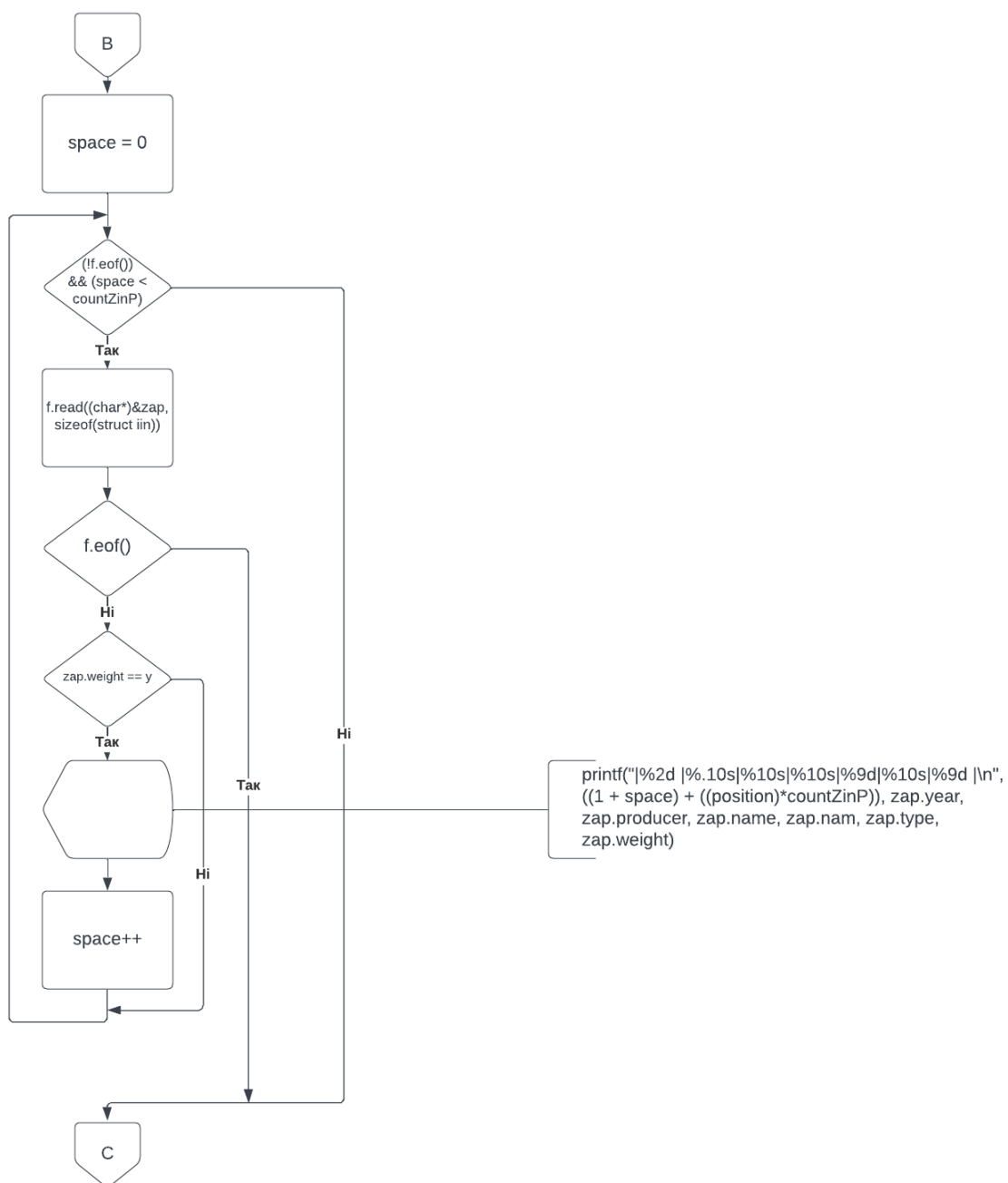


Рис 3.8. Блок-схема до процедури tsk3()

Розділ №4. Тексти програм

```
#include <iostream>
#include <fstream>
#include <conio.h>
#include <cstdlib>
#include <cstring>
#include <windows.h>
using namespace std;
HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE); /* Отримання
дескриптора пристрою стандартного виводу, а саме консолі */
//задаємо константи, що визначають довжину символічних полів: Виробник, Модель, Тип,
Рік випуску.
const int l_producer = 30, l_model = 30, l_type = 30, l_year = 10;
const int countZinP = 30; // оголошуємо константу, що регламентує кількість записів на
екрані
//основна структура
struct iin
{
    char producer[l_producer];
    char year[l_year];
    char name[l_model];
    int nam;
    int weight;
    char type[l_type];
};
// глобальні змінні
fstream f; // файлова змінна
char fdir[100] = "1.txt"; // змінна для збереження повного імені файлу
void menu(); // попереднє оголошення процедури
// *** Процедура створення/відкриття файлу ***
void open_new()
{
    system("cls");
    printf(" Вкажіть повне ім'я файлу ( Приклад: \"C:\\Program Files\\file.dat\" ): ");
    cin >> fdir;
opf:
    f.open(fdir, ios::in | ios::out | ios::binary); // відкриваємо файл у бінарному режимі
    if (!f.is_open()) //якщо файл не відкрився
    {
        printf("\n\n Помилка при відкритті файлу, буде спроба його створити.\n\n\n ");
        ofstream ofs(fdir); //створюємо файл
        ofs.close(); //закриваємо потік створеного файлу
        printf(" Файл створено. \n\n ");
        goto opf;
    }
    else
        printf(" Файл відкрито для роботи. \n\n ");
    system("pause");
}
// *** Процедура додавання запису ***
void addzap()
```

```

{
    system("cls");
    int i, // змінна призначена для номера нового запису
        pp, // змінна для збереження натиснутої клавіші
        j, // змінна для збереження довжини символьних змінних
        t, // змінна для збереження довжини символьних змінних
        q1;
    iin heli; // оголошуємо змінну для роботи із записом
    f.clear(); // очищуємо прапорці помилок
    f.seekg(0, ios::end); // переводимо вказівник на кінець файлу
    i = f.tellp() / sizeof(struct iin); // визначаємо кількість записів
s11:
    i++; // номер нового запису
    // обнулюємо поля запису
    strcpy(heli.year, "");
    strcpy(heli.producer, "");
    strcpy(heli.name, "");
    heli.nam = 0;
    heli.weight = 0;
    strcpy(heli.type, "");
    pp = 0;
    // далі заповнюємо поля нового запису
    cout << " Запис № " << i << ": \n";

    cout << " Рік випуску -> ";
    do
    {
        gets_s(heli.year); // зчитуємо значення у символьне поле
        t = strlen(heli.year); // визначаємо довжину символьного поля
    } while (t == 0); // цикл буде виконуватися до тих пір, поки довжина введенного
    // символьного поля буде нульовою
    for (j = t; j < l_year - 1; j++) // цикл забезпечує додавання до символьного поля пробілів
    // до заданої довжини
        strcat(heli.year, " ");

    cout << "\n Виробник -> ";
    do
    {
        gets_s(heli.producer); // зчитуємо значення у символьне поле
        t = strlen(heli.producer); // визначаємо довжину символьного поля
    } while (t == 0); // цикл буде виконуватися до тих пір, поки довжина введенного
    // символьного поля буде нульовою
    for (j = t; j < l_producer - 1; j++) // цикл забезпечує додавання до символьного поля
    // пробілів до заданої довжини
        strcat(heli.producer, " ");

    cout << "\n Модель -> ";
    do
    {
        gets_s(heli.name); // зчитуємо значення у символьне поле
        t = strlen(heli.name); // визначаємо довжину символьного поля
    }
}

```



```

    } while (t == 0); // цикл буде виконуватися до тих пір, поки довжина введеного
символьного поля буде нульовою
    for (j = t; j < l_model - 1; j++) // цикл забезпечує додавання до символьного поля пробілів
до заданої довжини
        strcat(heli.name, " ");

    cout << "\n Кількість пасажирських місць -> ";
    do
    {
        cin >> heli.nam; // зчитуємо значення у числове поле
    } while (t == 0); // цикл буде виконуватися до тих пір, поки довжина введеного
символьного поля буде нульовою

    cout << "\n Тип -> ";
    do
    {
        gets_s(heli.type); // зчитуємо значення у символьне поле
        t = strlen(heli.type); // визначаємо довжину символьного поля
    } while (t == 0); // цикл буде виконуватися до тих пір, поки довжина введеного
символьного поля буде нульовою
    for (j = t; j < l_type - 1; j++) // цикл забезпечує додавання до символьного поля пробілів
до заданої довжини
        strcat(heli.type, " ");

    cout << "\n Вантажопідйомність -> ";
    do
    {
        cin >> heli.weight; // зчитуємо значення у числове поле
    } while (t == 0); // цикл буде виконуватися до тих пір, поки довжина введеного
символьного поля буде нульовою

    f.write((char*)&heli, sizeof(struct iin)); // записуємо у файл сформований запис
s12:
    cout << " Для введення ще одного запису натисніть -> Enter, для закінчення -> Esc \n";
    if ((pp = _getch()) != 27)
        if (pp == 13)
            goto s11;
        else
            goto s12;
    }
// *** Процедура виведення усіх записів ***
void all_out()
{
    iin zap; // оголошуємо змінну для роботи із записом
    int space; // оголошуємо змінну, що є лічильником кількості записів на екрані
    int position = 0; // враховує к-сть виведених записів на сторінці
    int im = 0; // змінна для збереження коду натиснутої клавіші
    f.clear(); // очищуємо прапорці помилок
    f.seekp(0, ios::end); // переміщуємо вказівник на кінець файлу
    int size = f.tellp() / sizeof(struct iin); // визначаємо кількість записів у файлі
    if (size < 1)
    { // якщо записів немає, то дати повідомлення на екран

```



```

strcpy(heli.producer, "");
strcpy(heli.name, "");
heli.nam = 0;
heli.weight = 0;
strcpy(heli.type, "");
//далі заповнюємо поля нового запису
cout << "Запис № " << i << ": \n";
cout << "Рік випуску -> ";
do
{
    gets_s(heli.year);
    t = strlen(heli.year);
} while (t == 0);
for (j = t; j < l_year - 1; j++)
    strcat(heli.year, " ");

cout << "\n Виробник -> ";
do
{
    gets_s(heli.producer);
    t = strlen(heli.producer);
} while (t == 0);
for (j = t; j < l_producer - 1; j++)
    strcat(heli.producer, " ");

cout << "\n Модель -> ";
do
{
    gets_s(heli.name);
    t = strlen(heli.name);
} while (t == 0);
for (j = t; j < l_model - 1; j++)
    strcat(heli.name, " ");

cout << "\n Кількість пасажирських місць -> ";
do
{
    cin >> heli.nam;
} while (t == 0);

cout << "\n Тип -> ";
do
{
    gets_s(heli.type);
    t = strlen(heli.type);
} while (t == 0);
for (j = t; j < l_type - 1; j++)
    strcat(heli.type, " ");

cout << "\n Вантажопідйомність -> ";
do
{

```



```

        else
            goto st; // буде здійснено перехід на мітку st, якщо буде натиснута клавіша відмінна
від Enter та Esc
        if (f.is_open())
            f.close(); // закриваємо файл, якщо він був відкритий
        exit(0); // закриваємо програму / виходимо з неї
    ex::
}

// Порівняння двох рядків
int fyy(char* a, char* b)
{
    int i = 0, j = strlen(a);
    while (j > i)
    {
        if (a[i] != b[i])
            return 0;
        i++;
    }
    return 1;
}

// Процедура для виводу списку гелікоптерів за кількістю місць та типом
void tsk1()
{
    ll:
        system("cls");
        int y;
        char t[18];
        cout << "Введіть кількість пасажирських місць = ";
        cin >> y;
        cout << "Введіть тип гелікоптерів: ";
        cin >> t;

        iin zap; // оголошуємо змінну для роботи із записом
        int space; // оголошуємо змінну, що є лічильником кількості записів на екрані
        int position = 0; // враховує к-сть виведених записів на сторінці
        int im = 0; // змінна для збереження коду натиснутої клавіші
        f.clear(); // очищуємо прапорці помилок
        f.seekp(position * countZinP * sizeof(struct iin), ios::beg); // переміщуємо вказівник на
потрібний запис, що враховується за наступною формулою
        position* countZinP * sizeof(struct iin);
        cout <<
        "_____
        _____
        _____.\n";
        cout << "| № | Рік випуску | Виробник | Модель | Кількість
пасажирських місць | Тип | Вантажопідйомність КГ | \n";
        cout <<
        "===|=====|=====|=====|=====
        =====|=====|=====|=====
        =====\n";
        space = 0; //обнуляємо значення лічильника кількості записів на екрані

```



```

        goto ex;
    }
    else
    {
        goto l2;
    }
}
ex:
    system("pause");
}
//Процедура виводу списку гелікоптерів гелікоптерів заданого виробника
void tsk4()
{
    ll:
        system("cls");
        char y[35];
        cout << "Введіть назву виробника = ";
        cin >> y;
        iin zap; // оголошуємо змінну для роботи із записом
        int space; // оголошуємо змінну, що є лічильником кількості записів на екрані
        int position = 0; // враховує к-сть виведених записів на сторінці
        int im = 0; // змінна для збереження коду натиснутої клавіші
        f.clear(); // очищуємо прапорці помилок
        f.seekp(position * countZinP * sizeof(struct iin), ios::beg); // переміщуємо вказівник на
        потрібний запис, що враховується за наступною формулою
        position* countZinP * sizeof(struct iin);
        cout <<
        "_____
        _____
        _____.\n";
        cout << "| № | Рік випуску | Виробник | Модель | Кількість
        пасажирських місць | Тип | Вантажопідйомність КГ | \n";
        cout <<
        "=====|=====|=====|=====|=====
        =====|=====|=====|=====|=====
        =====|\n";
        space = 0; //обнуляємо значення лічильника кількості записів на екрані
        int a, b, c, a1, b1, c1;
        while ((!f.eof()) && (space < countZinP)) //умова виконання циклу: поки не досягли кінця
        файлу або кількість виведених записів на екран менше countZinP
        {
            f.read((char*)&zap, sizeof(struct iin)); //зчитування запису з файлу у змінну zap
            if (f.eof()) break; // якщо досягнуто кінець файлу, то вийти з циклу
            // виводимо поля зчитаного запису на екран
            if (fyu(y, zap.producer) == 1)
            {
                printf("%2d |%.10s |%10s |%10s |%9d |%10s|%9d | \n", ((1 +
                space) + ((position)*countZinP)), zap.year, zap.producer, zap.name, zap.nam, zap.type,
                zap.weight);
            }
            space++;
        }
}

```



```

        .\n";
        cout << " № | Рік випуску | Виробник | Модель | Кількість
пасажирських місць | Тип | Вантажопідйомність КГ | \n";
        cout <<
        "=====|=====|=====|=====|=====
=====|=====|=====|=====|=====
=====|\n";

        int position = 0;
        int space = 0;
        iin zap;

        f.clear();
        f.seekp(position * countZinP * sizeof(struct iin), ios::beg);
        position * countZinP * sizeof(struct iin);

        bool dataFound = false; // Додано змінну для перевірки, чи були знайдені дані в проміжку

        while ((!f.eof()) && (space < countZinP))
        {
            f.read((char*)&zap, sizeof(struct iin));
            if (f.eof()) break;

            int year = stringToInt(zap.year, 0, 4);
            if (xValue <= year && year <= yValue)
            {
                // Запис потрапляє в проміжок, не виводимо його
                continue;
            }

            dataFound = true; // Знайдено дані поза проміжком
            printf("%2d |%.10s |%10s |%10s |%9d |%10s|%9d | \n", ((1 +
space) + ((position)*countZinP)), zap.year, zap.producer, zap.name, zap.nam, zap.type,
zap.weight);

            space++;
        }
        cout <<
        "=====|=====|=====|=====|=====
=====|=====|=====|=====|=====
=====|\n";

        if (!dataFound)
        {
            cout << "Дані поза заданим проміжком не знайдені.\n";
        }

        cout << "\n Натисніть: \n\n Enter для того, щоб зробити перевірку заново. \n\n Esc для
того, щоб повернутися в головне меню. \n\n\n";
        l2:
        int go = _getch(); //зчитуємо натиснуту клавішу

```

```

if (go == 13) // перевірка чи не натиснута клавіша Enter
{
    goto l1;
}
if (go == 27) // перевірка чи натиснута клавіша Esc
{
    goto ex;
}
else
{
    goto l2;
}
ex:
    system("pause");
}
//Процедура виводу списоку гелікоптерів, які випустили у заданий період
void tsk6()
{
l1:
    system("cls");
    char x[5], y[5];
    cout << "Введіть початковий рік: ";
    cin >> x;
    cout << "Введіть кінцевий рік: ";
    cin >> y;
    int xValue = stringToInt(x, 0, 4);
    int yValue = stringToInt(y, 0, 4);

    cout << "Результати за період " << xValue << " - " << yValue << ":\n";
    cout <<
    "._____.\n";
    cout << "| № | Рік випуску | Виробник | Модель | Кількість\n";
    cout << "пасажирських місць | Тип | Вантажопідйомність КГ | \n";
    cout <<
    "||=====||=====||=====||=====||\n";
    cout <<
    "=====||=====||=====||=====||\n";

    int position = 0;
    int space = 0;
    iin zap;

    f.clear();
    f.seekp(position * countZinP * sizeof(struct iin), ios::beg);
    position * countZinP * sizeof(struct iin);

    while ((!f.eof()) && (space < countZinP))
    {
        f.read((char*)&zap, sizeof(struct iin));
        if (f.eof()) break;
    }

```



```

cout << " | 6. Група: КН-1-2                                     |\n";
cout << "
=====
=====|\n";
cout << "\n Натисніть: \n\n Esc для того, щоб повернутися в головне меню. \n\n\n";
12:
int go = _getch(); //зчитуємо натиснуту клавішу
if (go == 13) // перевірка чи не натиснута клавіша Enter
{
    goto 11;
}
if (go == 27) // перевірка чи натиснута клавіша Esc
{
    goto ex;
}
else
{
    goto 12;
}
ex:
system("pause");
}
// *** Реалізація текстового інтерфейсу користувача ***
void menu()
{
    int im = 0, p = 0; //змінні для збереження кодів натиснутих клавіш при роботі з пунктами
меню
COORD crd, end; //оголошуємо змінні типу координат, що мають поле X та Y
    crd.X = 2; // задаємо початкову позицію курсору по осі X для відображення знаку
вибору
    crd.Y = 5; // задаємо початкову позицію курсору по осі Y для відображення знаку
вибору
    end.X = 0; // задаємо позицію курсору по осі X за межами меню
    end.Y = 15; // задаємо позицію курсору по осі Y за межами меню
start: // початок виведення меню на екран
    system("cls");
    cout << "
=====
=====|\n";
    cout << " |                                \"Довідник гелікоптерів\"                                |\n";
    cout << "
|*****|
\n";
    cout << " | * Головне меню *                                     |\n";
    cout << "
|*****|
\n";
    cout << " | 1. Відкрити/створити файл                                     |\n";
    cout << " | 2. Додати запис у файл                                     |\n";
    cout << " | 3. Редагувати запис                                         |\n";
    cout << " | 4. Видалити запис з файлу                                    |\n";
    cout << " | 5. Вивести усю інформацію з файлу                            |\n";

```



```

    cout << " | 6. Список гелікоптерів із заданими кількістю пасажирських місць і типу
\n";
    cout << " | 7. Список гелікоптерів гелікоптерів заданого типу                \n";
    cout << " | 8. Список гелікоптерів із заданою вантажопідємністю                \n";
    cout << " | 9. Список гелікоптерів заданого виробника                        \n";
    cout << " | 10. Список гелікоптерів, які випустили до/після заданого року        \n";
    cout << " | 11. Список гелікоптерів, які випустили у заданий період            \n";
    cout << " | 12. Інформація про розробника                                     \n";
    cout << " | 13. Вихід                                                         \n";
    cout << "
=====
===== \n";
    if (!f.is_open()) // в залежності від того чи відкрито файл даємо відповідне повідомлення
        cout << "\n Файл відсутній, завантажте або створіть файл. \n";
    else
        cout << "\n Файл успішно завантажено.: \"\" << fdir << "\"\n";
    SetConsoleCursorPosition(hConsole, crd); // переводимо курсор в позицію координат crd
    SetConsoleTextAttribute(hConsole, (WORD)((20 << 0) | 10)); //задаємо колір шрифту та
фону для виведення
    cout << ">>>";
    SetConsoleTextAttribute(hConsole, (WORD)((15 << 0) | 0)); //задаємо колір шрифту та
фону для виведення
    SetConsoleCursorPosition(hConsole, end); //встановлюємо курсор у нижній лівий кут
    im = _getch(); //отримуємо код натиснутої клавіші
    switch (im)
    {
    case 72: // якщо натиснута клавіша "стрілка у гору" - Up
    {
        if (crd.Y > 5) crd.Y--; // якщо не досягнуто перший пункт меню, то зменшуємо
значення координат crd.Y на 1 позицію
        goto start; // переходимо на мітку start
    }
    case 80: // якщо натиснута клавіша "стрілка вниз" - Down
    {
        if (crd.Y < 17) crd.Y++; // якщо не досягнуто останній пункт меню, то збільшуємо
значення координат crd.Y на 1 позицію
        goto start; // переходимо на мітку start
    }
    case 13:
    {
        p = crd.Y + 45; // обраховуємо номер пункту меню, що буде в межах
        if ((!f.is_open()) && (p != 61)) p = 50; // якщо файл не відкрито, то буде здійснено
перехід до пункту відкриття файлу
        switch (p)
        {
        case 50: // якщо обрано перший пункт меню
        {
            open_new(); // викликаємо функцію відкриття файл
            goto start; // переходимо на мітку start
        }
        case 51:
        {

```

```

    addzap(); // викликаємо функцію додавання записів у файл
    goto start; // переходимо на мітку start
}
case 52:
{
    editzap(); // викликаємо функцію редагування записів у файл
    goto start; // переходимо на мітку start
}
case 53:
{
    delzap(); // викликаємо функцію видалення запису з файлу
    goto start; // переходимо на мітку start
}
case 54:
{
    all_out(); // викликаємо функцію виведення усіх записів з файлу
    goto start; // переходимо на мітку start
}
case 55:
{
    tsk1();
    goto start; // переходимо на мітку start
}
case 56:
{
    tsk2();
    goto start; // переходимо на мітку start
}
case 57:
{
    tsk3();
    goto start; // переходимо на мітку start
}
case 58:
{
    tsk4();
    goto start; // переходимо на мітку start
}
case 59:
{
    tsk5();
    goto start; // переходимо на мітку start
}
case 60:
{
    tsk6();
    goto start; // переходимо на мітку start
}
case 61:
{
    info();
    goto start; // переходимо на мітку start
}

```

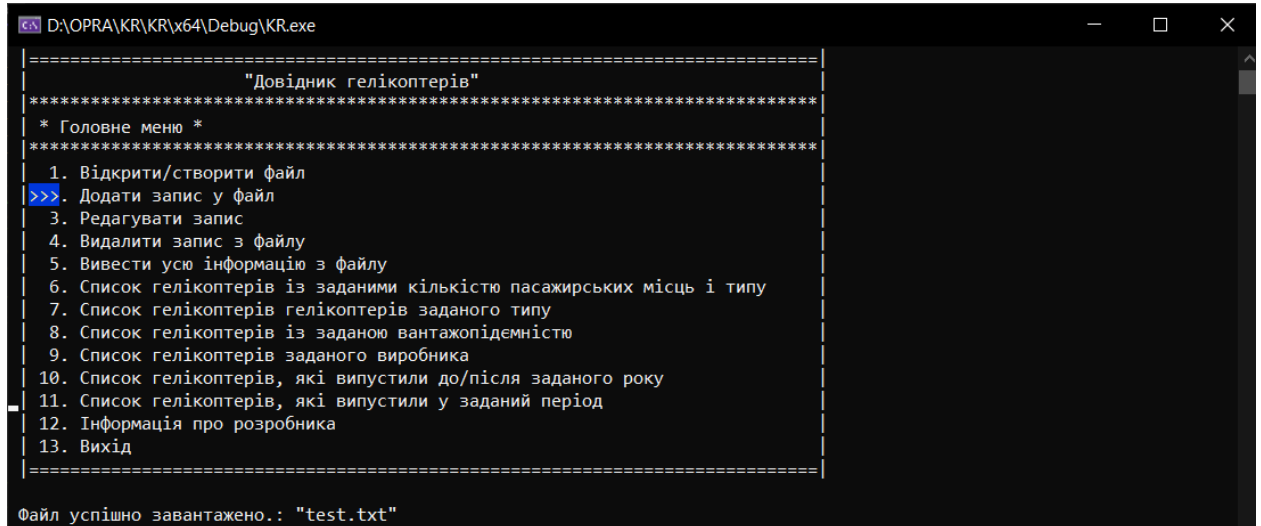
```

    }
    case 62:
    {
        exit_prog();// викликаємо функцію завершення виконання проєкту
        goto start; // переходимо на мітку start
    }
    default: goto start; // переходимо на мітку start
    }
}
default: goto start;
}
}
//Головна функція
int main()
{
    SetConsoleCP(1251); //встановлюємо 1251 кодування для шрифту консолі
    SetConsoleOutputCP(1251);
    system("color 07"); //встановлюємо колір фону консолі чорним, а текст - білим
    menu(); // викликаємо процедуру menu
    system("pause");
    return 0;
}

```

Розділ №5. Інструкція користувача

У вашій папці з кодом вже є два файли: "heli.txt" і "heli.dat". Обидва файли мають однаковий вміст, але "heli.txt" був створений як резервна копія. Ви можете використовувати ці файли безпосередньо, не видаляючи їх з папки. Щоб застосувати ці файли, оберіть пункт "1" (Рис 5.1) і введіть "heli.dat" або "heli.txt".



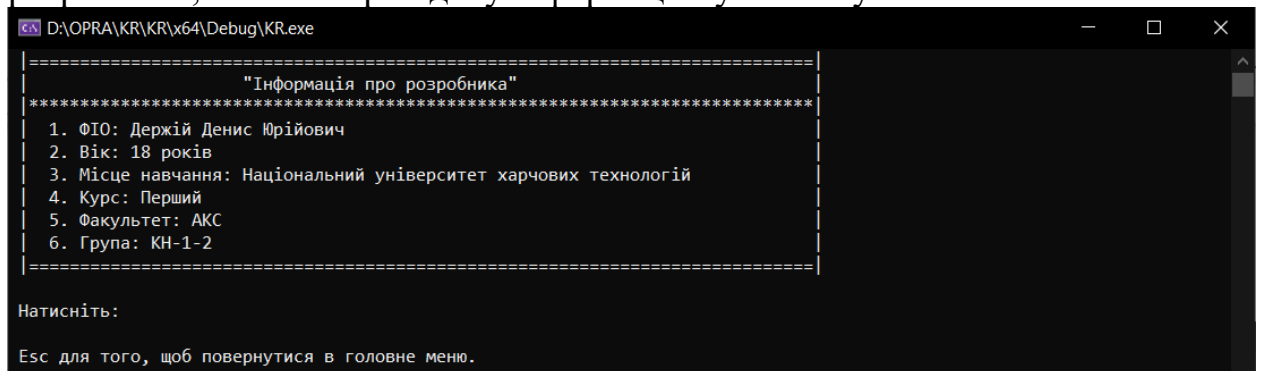
```
D:\OPRA\KR\KR\x64\Debug\KR.exe

=====
"Довідник гелікоптерів"
=====
* Головне меню *
=====
1. Відкрити/створити файл
2. Додати запис у файл
3. Редагувати запис
4. Видалити запис з файлу
5. Вивести усю інформацію з файлу
6. Список гелікоптерів із заданими кількістю пасажирських місць і типу
7. Список гелікоптерів заданого типу
8. Список гелікоптерів із заданою вантажопідємністю
9. Список гелікоптерів заданого виробника
10. Список гелікоптерів, які випустили до/після заданого року
11. Список гелікоптерів, які випустили у заданий період
12. Інформація про розробника
13. Вихід
=====

Файл успішно завантажено.: "test.txt"
```

Рис 5.1. Головне меню

Обравши пункт "12" (Рис 5.2), ви можете дізнатися всю інформацію про розробника, який створив дану інформаційну систему.



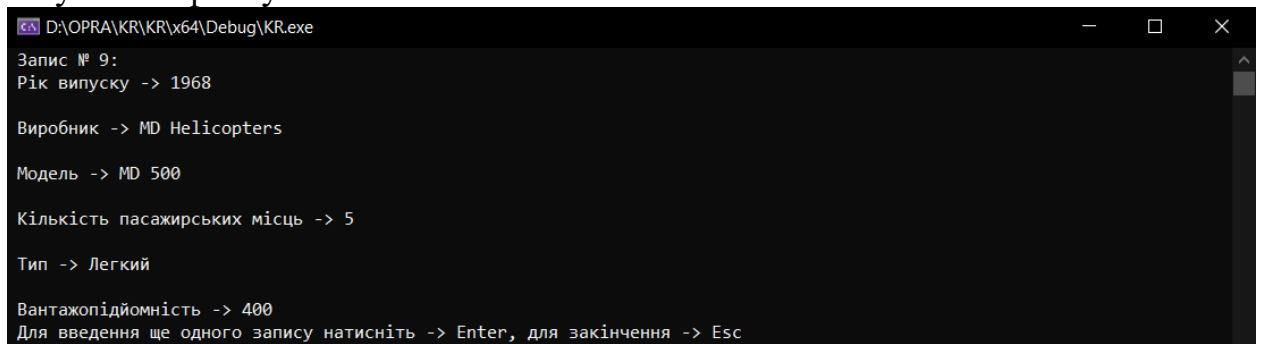
```
D:\OPRA\KR\KR\x64\Debug\KR.exe

=====
"Інформація про розробника"
=====
1. ФІО: Держій Денис Юрійович
2. Вік: 18 років
3. Місце навчання: Національний університет харчових технологій
4. Курс: Перший
5. Факультет: АКС
6. Група: КН-1-2
=====

Натисніть:
Esc для того, щоб повернутися в головне меню.
```

Рис 5.2. Інформація про розробника

Обравши пункт "2" (Рис 5.3), ви можете додати новий запис до уже існуючого файлу.



```
D:\OPRA\KR\KR\x64\Debug\KR.exe

Запис № 9:
Рік випуску -> 1968

Виробник -> MD Helicopters

Модель -> MD 500

Кількість пасажирських місць -> 5

Тип -> Легкий

Вантажопідємність -> 400

Для введення ще одного запису натисніть -> Enter, для закінчення -> Esc
```

Рис 5.3. Додавання запису

Обравши пункт “3” (Рис 5.4), ви можете редагувати будь-який запис з файлу.

D:\OPRA\KR\KR\64\Debug\KReke

<<< Загальна кількість записів у файлі = 13 >>>

№	Рік випуску	Виробник	Модель	Кількість пасажирських місць	Тип	Вантажопідйомність КГ
1	1996	Airbus Helicopters	HI25	6	Багатоцільовий	1300
2	1996	Bell Helicopter	Bell 407	6	Багатоцільовий	1300
3	1992	Robinson Helicopter	R44	4	Легкий	600
4	2004	Sikorsky Aircraft	S-92	19	Пасажирський	5400
5	2003	Airbus Helicopters	Eurocopter Tiger	2	Атакуючий	1200
6	1979	Bell Helicopter	Bell 412	15	Середня	3000
7	1973	Kamov	Ka-27	14	Важкий	4500
8	2010	Robinson Helicopter	R66	4	Легкий	400
9	1968	MD Helicopters	MD 500	5	Легкий	400
10	1997	Kamov	Ka-52 Alligator	2	Бойовий	2000
11	1974	Sikorsky Aircraft	S-70 Black Hawk	10	Важкий	4500
12	1979	Robinson Helicopter	R22	2	Легкий	200
13	1959	Bell Helicopter	UH-1 Iroquois	10	Середня	1100

<<< Up/Down - переміщення на стор. <<< № 1 >>> Esc - головне меню >>>

Рис 5.4. Вивід всієї інформації

Обравши пункт “4” (Рис 5.5), ви можете видалити будь-який запис з файлу.

D:\OPRA\KR\KR\64\Debug\KReke

Введіть номер запису, що буде відредаговано -> 5

№	Рік випуску	Виробник	Модель	Кількість пасажирських місць	Тип	Вантажопідйомність КГ
5	2003	Airbus Helicopters	Eurocopter Tiger	2	Атакуючий	1200

<<<<<<< Для підтвердження редагування натисніть Enter, а для відміни Esc >>>>>>>

Запис № 0:

Рік випуску -> 2002

Виробник -> Airbus Helicopters

Модель -> Eurocopter Tiger

Кількість пасажирських місць -> 2

Тип -> Атакуючий

Вантажопідйомність -> 1300

<<<<<<< Для підтвердження редагування натисніть Enter, а для відміни Esc >>>>>>>

Рис 5.5. Редагування запису

Обравши пункт “5” (Рис 5.6), ви можете вивести усю інформацію з файлу.

D:\OPRA\KR\KR\64\Debug\KReke

Введіть номер запису, що буде видалено з файлу -> 5

Цей запис буде видалено з файлу

№	Рік випуску	Виробник	Модель	Кількість пасажирських місць	Тип	Вантажопідйомність КГ
5	2002	Airbus Helicopters	Eurocopter Tiger	2	Атакуючий	1300

<<< Для підтвердження редагування натисніть Enter, а для відміни Esc >>>

Рис 5.6. Видалення запису

Обравши пункти від “6” по “11” (Рис 5.7 - 5.12), вам буде надана можливість вивести список із записів, за вказаним параметром.

D:\OPRA\KR\KR\64\Debug\KReke

Введіть тип гелікоптерів: Легкий

№	Рік випуску	Виробник	Модель	Кількість пасажирських місць	Тип	Вантажопідйомність КГ
3	1992	Robinson Helicopter	R44	4	Легкий	600
7	2010	Robinson Helicopter	R66	4	Легкий	400
8	1968	MD Helicopters	MD 500	5	Легкий	400
11	1979	Robinson Helicopter	R22	2	Легкий	200

Натисніть:

Enter для того, щоб зробити перевірку заново.

Esc для того, щоб повернутися в головне меню.

Рис 5.7. Сортування за типом гелікоптерів

D:\OPRA\KR\KR\64\Debug\KReke

Введіть вантажопідйомність = 4500

№	Рік випуску	Виробник	Модель	Кількість пасажирських місць	Тип	Вантажопідйомність КГ
6	1973	Kamov	Ka-27	14	Важкий	4500
10	1974	Sikorsky Aircraft	S-70 Black Hawk	10	Важкий	4500

Натисніть:

Enter для того, щоб зробити перевірку заново.

Esc для того, щоб повернутися в головне меню.

Рис 5.8. Сортування за вантажопідйомністю гелікоптерів

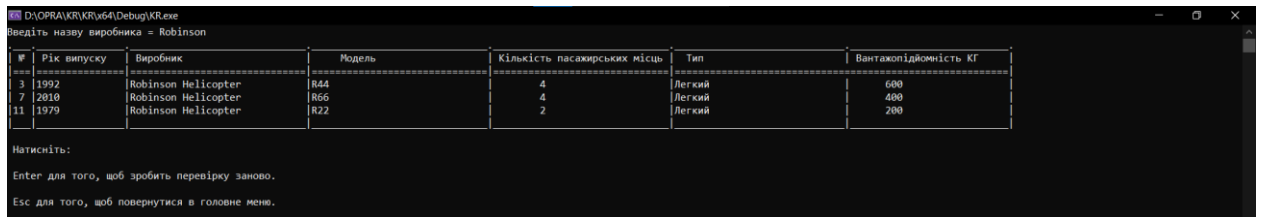


Рис 5.9. Сортування за виробником гелікоптерів

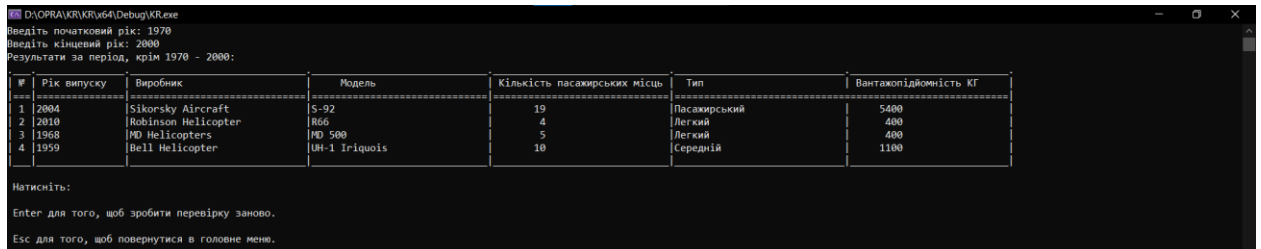


Рис 5.10. Сортування гелікоптерів, які випустили до/після заданого року

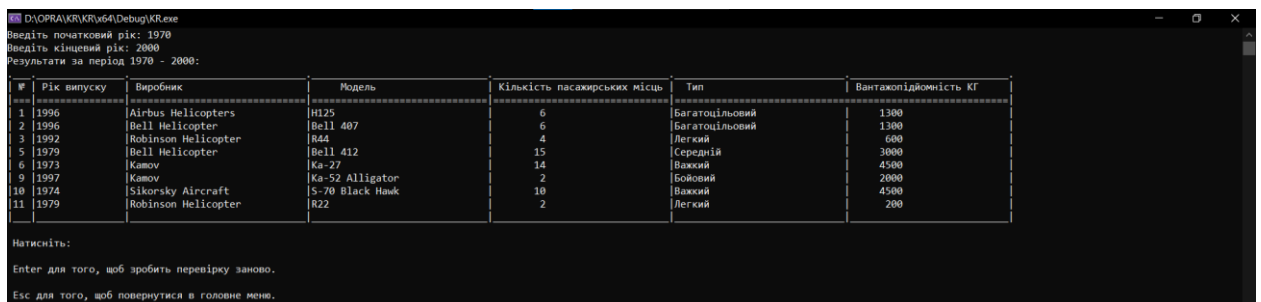


Рис 5.11. Сортування гелікоптерів, які випустили у заданий період

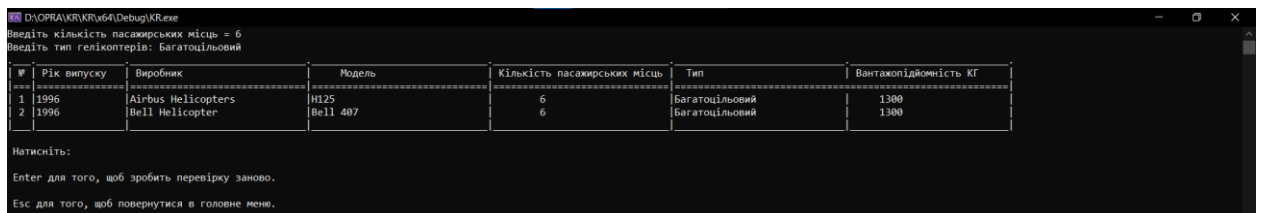


Рис 5.12. Сортування гелікоптерів із заданими кількістю пасажирських місць і типу

Обравши пункт “13” (Рис 5.13), ви можете завершити роботу з програмою, при цьому всі зміни будуть збережені автоматично.

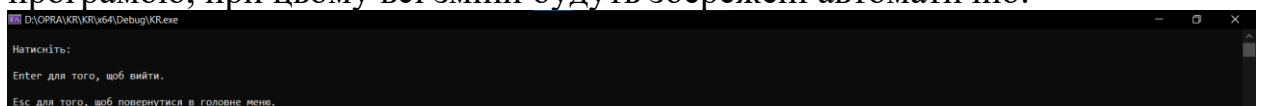


Рис 5.13. Вихід з програми

Список використаних джерел

Книга:

1. Герберт Ш. Повний довідник із C++ – перекладено з англійської: Видавничий дім “Вільямс”, 2006. – 800 с
2. Ярмуш, О. В. Інформатика і комп’ютерна техніка : навч. посіб. / О. В. Ярмуш, М. М. Редько. – К. : Вища освіта, 2006. – 359 с. Деннис, М. Р. Язык программирования C. / М. Р. Деннис, У. К. Брайан ; пер. с англ. – М. : «Вильямс», 2009.
3. Васильєв, О. Програмування C++ в прикладах і задачах. Навч. пос. Збільшений формат В5 [Текст] / О. Васильєв. – К. : Ліра-К, 2020.

Електронні ресурси:

1. Вільна енциклопедія – Режим доступу до ресурсу:
<https://uk.wikipedia.org/wiki/>.
2. Гелікоптери-Мілітарний – Режим доступу до ресурсу:
<https://mil.in.ua/uk/tag/helikoptery/>.
3. Вертолїт – Режим доступу до ресурсу:
<https://www.ukrinform.ua/tag-vertolit>.
4. aCode – Режим доступу до ресурсу:
<https://acode.com.ua>.
5. Військовий портал Defense Express – Режим доступу до ресурсу:
<https://defence-ua.com>.
6. АрміяInform – Режим доступу до ресурсу:
<https://armyinform.com.ua/tag/gelikoptery/>.
7. Stack Overflow – Режим доступу до ресурсу:
<https://stackoverflow.com>.
8. C++ Reference – Режим доступу до ресурсу:
<https://en.cppreference.com/w/cpp>.