ВСТУП

Методичні вказівки для студентів другого курсу за напрямом підготовки 6.050102 «Комп'ютерна інженерія» призначені для набуття практичних навиків під година розробки алгоритмів та написання програм на мові С++, яка вивчається в навчальній дисципліні «Об'єктно-орієнтоване програмування».

Основною метою лабораторних робіт є:

- опанування алгоритмічної мови програмування С++;
- набуття навичок у розробці об'єктно-орієнтованих програм із застосуванням механізмів інкапсуляції, наслідування та поліморфізму;

Структура лабораторних робіт включає мету роботи, короткі теоретичні відомості, методичні вказівки для підготування до роботи та її виконання, індивідуальні завдання до лабораторної роботи, контрольні запитання для підготування до захисту роботи. Тематика лабораторних робіт охоплює практично всі аспекти об'єктно-орієнтованого програмування на мові С++.

Для захисту виконаної лабораторної роботи студентові необхідно підготувати звіт роботи, який винний включати:

1) Титульний аркуш із вказанням назви лабораторної роботи, тими лабораторної роботи.

Наступні листи повинні містити:

2) Завдання для лабораторної роботи.
3) Результати роботи програми.
4) Висновок, який включає в собі опис виконаної роботи із зазначенням «***що зроблено?», «за допомогою чого це зроблено?», «які отримано результати?», «вірність отриманих результатів»***.

Приклад титульного аркуша звіту наведений у додатку А.

Завдання до кожної лабораторної роботи мають три рівні складності, яки кожен студент обирає на власний розсуд. Рівні складності визначають рівень можливої максимальної оцінки за роботові, що виконується.

ЛАБОРАТОРНА РОБОТА №1
Основні відмінності мови програмування С++ від Сі

Позначка: вивчити структуру програми мови С++, навчитись працювати з середовищами програмування MS Microsoft Visual Studio, ознайомитися з основними відмінностями операторів введення-виведення, організації функцій, коментарів, отримати навички з використання механізму перевантаження змінних та функцій, отримати навички з використання файлових операцій в мові С++.

Теоретичні відомості

Короткий огляд С++

С++ - це розширена версія мови С. С++ містить у собі все, що є в Сі, але крім цього він підтримує об'єктно-орієнтоване програмування (Object Oriented Programming, ООР), у С++ є безліч додаткових можливостей, які незалежно від об'єктно-орієнтованого програмування роблять його простішою мовою.

Оскільки С++ будується на мові Сі, то структура програми на С++ ідентична за відмінністю деяких деталей. Зокрема інструкція #include підключає до програми заголовочний файл iostream.h, що забезпечує підтримку системи введення/виведення С++. (У С++ цей файл має те ж саме призначення, що й файл stdio.h у С.)

Нижче представлена друга версія програми, у якій використається сучасний стиль. Приклад прогами на С++ у сучасному стилі використовує нове оформлення заголовків і ключове слово namespace:

```
#include <iostream>
using namespace std;
void main ()
{
// програмний код
}
```

В інструкції #include після слова iostream відсутні символи .h. По-друге, у наступному рядку задається так називаний простір імен (namespace).

Нові заголовки в програмах на С++

Як вам повинне бути відомо з досвіду програмування на Сі, при використанні бібліотечної функції в програму необхідно включити заголовочний файл. Це робиться за допомогою інструкції #include <ім'я_біліотеки>. Наприклад, при написанні програм мовою

Сі заголовочним файлом для функцій введення/виведення є файл stdio.h, що включається в програму за допомогою наступної інструкції:

```
#include <stdio.h>
```

У перші кілька років після появи C++ у ньому використався той же стиль оформлення заголовків, що й у C. Для сумісності з колишніми програмами в мові Standard C++ цей стиль як і раніше підтримується. Проте при роботі з бібліотекою Standard C++ відповідно до нового стилю замість імен заголовних файлів указуються стандартні ідентифікатори, зя якими компілятор знаходить необхідні файли. Нові заголовки C++ є абстракціями, що гарантують оголошення відповідних прототипів і визначень бібліотеки мови Standard C++.

Оскільки нові заголовки не є іменами файлів, для них не потрібно вказувати розширення .h, а тільки ім'я заголовка в кутових дужках. Нижче представлені кілька заголовків, підтримуваних у мові Standard C++:

```
<iostream>
<fstxeam>
<vector>
<string>
```

Такі заголовки як і раніше включаються в програму за допомогою інструкції #include. Єдиною відмінністю є те, що нові заголовки зовсім не обов'язково є іменами файлів.

Оскільки C++ містить всю бібліотеку функцій Сі, як і раніше підтримується стандартний стиль оформлення заголовних файлів бібліотеки C. Таким чином, такі заголовні файли, як stdio.h й ctype.h усе ще доступні. Однак Standard C++ також визначає заголовки нового стилю, які можна вказувати замість цих заголовних файлів. Відповідно до версії C++ до стандартних заголовків Сі просто додається префікс із і відділяється розширення .h. Наприклад, заголовок math.h заміняється новим заголовком C++ <cmath>, а заголовок string.h - заголовком <cstring>. Хоча в цей час при роботі з функціями бібліотеки Сі допускається включати в програми заголовні файли у відповідності зі стилем Сі, такий підхід не схвалюється стандартом мови Standard C++. (Тобто, він не рекомендується.)

Оскільки заголовки нового стилю з'явилися в C++ зовсім недавно, у багатьох і багатьох колишніх програмах ви їх не знайдете. У цих програмах у відповідності зі стилем Сі у заголовках зазначені імена файлів.

Всі компілятори C++ підтримують заголовки старого стилю. Проте такі заголовки оголошені застарілими й не рекомендуються.

Простір імен

Коли ви включаєте в програму заголовок нового стилю, зміст цього заголовка виявляється в просторі імен std. Простір імен (namespace) - це просто певна оголошувана область, що, необхідна для

того, щоб уникнути конфліктів імен ідентифікаторів. Традиційно імена бібліотечних функцій й інших подібних ідентифікаторів розташовувалися в глобальному просторі імен (як, наприклад, у Сі). Однак зміст заголовків нового стилю міститься в просторі імен std. Для того, щоб простір імен std став видимим, необхідно використати наступну інструкцію:

```
using namespace std;
```

Ця інструкція розміщує std у глобальний простір імен. Після того як компілятор обробить цю інструкцію, стає можливим працювати із заголовками як старого, так і нового стилю.

Консольне введення і виведення в мові С++

Відмінною рисою мови С++ від і є підхід до введення і виведення. Хоча такі функції, як printf() і scanf, як і раніше доступні, С++ забезпечує інший, кращий спосіб виконання цих операцій. У С++ введення/виведення виконується з використанням операторів, а не функцій введення/виведення. Оператор виводу - це <<, а оператор введення - >>. Розглянемо наступну інструкцію С++:

cout<<"Цей рядок виводиться на екран";

Ця інструкція здійснює виведення рядка в заздалегідь визначений потік cout, що автоматично зв'язується з терміналом, коли програма С++ починає виконуватися. Це нагадує дію функції stdout у мові С. Як й у Сі, термінал для введення/виведення в С++ може бути перевизначений, але поки будемо вважати, що використається екран.

За допомогою оператора виводу << можна вивести дані будь-якого базового типу С++. Наприклад, інструкція здійснює виведення величини 100.99:

```
cout << 100.99;
```

У загальному випадку, для виводу на екран термінала використається наступна звичайна форма оператора <<:

```
cout << вираз;
```

Тут вираз може бути будь-яким дійсним вираженням С++, включаючи інші вираз виводу.

Для зчитування значення із клавіатури використовується оператор введення >>. Наприклад, у цьому фрагменті ціла величина вводиться в num:

```
int num;
cin >> num;
```

Зверніть увагу, що змінної num не передує амперсанд & Відомо, що при уведенні з використанням функції scanf мови Сі їй повинні передаватися адреси змінних. У випадку використання оператора введення С++ все відбувається інакше.

У загальному випадку для введення значення із клавіатури, використайте наступну форму оператора >>:

```
cin >> змінна;
```

Для правильного використання операторів введення/виведення в С++ необхідно включити в програму заголовочний файл iostream. Він є одним зі стандартних заголовних файлів С++ і поставляється з компілятором.

Примітка1. Для коректного виведення даних, представлених у кирилиці для середовища програмування MS Microsoft VisualStudio, необхідно на початку програми помістит команду

```
setlocale(LC_ALL, "Ukrainian");
```

Примітка 2. Для того, щоб можна було побачити результати роботи програм в консолі (середовище програмування MS Microsoft VisualStudio), необхідно перед завершенням основної функції помістити команду

```
cin.get();
```

Приклад1. У програмі виводиться рядок, два цілих числа й одне число із плаваючою крапкою подвійної точності:

```
#include <iostream>
using namespace std;
void main()
{
    setlocale(LC_ALL, "Ukrainian");
    int i, j;
    double d;
    i = 10;
    j = 20;
    d = 99.101;
    cout << "Ось кілька чисел: ";
    cout << i;
    cout << ' ';
    cout << j;
    cout << ' ';
    cout << d;
cin.get();
}
```

Нижче представлений результат роботи програми: Ось кілька чисел: 10 20 99.101.

Приклад2. В одному виразі введення/виведення можна виводити більше однієї величини. Наприклад, версія програми, описаної в прикладі 1, показує один з ефективних способів програмування інструкцій введення/виведення:

```
#include <iostream>
using namespace std;
void main()
{
```

```
    setlocale(LC_ALL, "Ukrainian");
    int i, j;
    double d;
    i = 10;
    j = 20;
    d = 99.101;
    cout << "Ось кілька чисел: ";
    cout << i << ' ' << j << ' ' << d;
    cin.get();
      }
```

Виведення здійснюється для декількох елементів даних в одному виразі. У загальному випадку ви можете використати єдину інструкцію для виводу будь-якого необхідної кількості елементів даних. Якщо це здається незручним, просто запам'ятаєте, що оператор виводу << поводиться так само, як і будь-який інший оператор C++, і може бути частиною довільно довгого виразу.

Варто включати в програму пробіли між елементами даних. Якщо пробілів не буде, то дані, виведені на екран, буде незручно читати. Ця програма пропонує користувачеві ввести ціле число:

```
#include <iostream>
using namespace std;
void main()
{
    setlocale(LC_ALL, "Ukrainian");
    int i;
    cout << "Введіть число: ";
    cin >> i;
    cout << "Ось ваше число: " << i << endl;
    cin.get();
      }
```

Результат роботи програми:

Введіть число: 100

Ось ваше число: 100

Наступний приклад введення цілого, числа із плаваючою крапкою й рядка символів. У ній для введення всього перерахованого використається одна інструкція.

```
#include <iostream>
using namespace std;
void main()
{
    setlocale(LC_ALL, "Ukrainian");
    int i;
    float f;
    char s[80];
```

```
    cout << "Введіть ціле, число с плаваючою
крапкою і рядок: ";
    cin >> i >> f >> s;
    cout << "Ось аши дані: ";
    cout << i << ' ' << f << ' ' << s;
    cin.get();
      cin.get();
      }
```

Як видно з прикладу, можна ввести в одній інструкції введення стільки елементів даних, скільки потрібно.

За замовчуванням при використанні оператора >> буферизується все введення рядка. Це означає, що доти, поки не буде натиснуто клавішу <Enter> інформація не буде передана у вашу програму. Розглянемо наступну програму:

```
#include <iostream>
using namespace std;
void main()
{
   setlocale(LC_ALL, "Ukrainian");
   char ch;
   cout <<"Вводьте символи поки не символ x <<
endl;
    do {
        cout << ":";
        cin >> ch;
    } while (ch !='x');
      cin.get();
      }
```

Для зчитування кожного чергового символу необхідно натискати клавішу <Enter>.

Коментарі в С++

У С++ коментарі в програму можна включати двома різними способами. Перший спосіб - це використання стандартного механізму, такого ж, як у Сі, тобто коментар починається з /* і кінчається */ Як і у Сі, у С++ цей тип коментарю не може бути вкладеним.

Другим способом, яким ви можете писати коментарі в програмах С++, є однорядковий коментар. Однорядковий коментар починається із символів // і закінчується кінцем рядка. Іншого символу, крім фізичного кінця рядка (такого, як повернення каретки/переклад рядка), в однорядковому коментарі не використається.

Приклад !!. Програма, у якій є стилі коментарів як Сі, так і С++:

/* Цей коментар у стилі С. Дана програма визначає парність цілого */

```
#include <iostream>
using namespace std;
void main()
{
    setlocale(LC_ALL, "Ukrainian");
    int num; // це однорядковий коментар С++
             // читання числа
    cout << "Уведіть перевіряє число, що:";
    cin >> num;
// перевірка на парність
    if ((num % 2) == 0)
            cout << "Число парне"<<endl;
            else cout <<"Число непарне" <<endl;
      cin.get();
      }
```

Деякі відмінності мов С и С++

У мови С++ є ряд невеликих, але важливих відмінностей від мови Сі.

1. Якщо в мові Сі функція не має параметрів, її прототип містить слово void у списку параметрів функції. Наприклад, якщо в Сі функція fl() не має параметрів (і повертає char), її прототип буде виглядати в такий спосіб:

```
char f (void);
```

У С++ слово void не обов'язково. Тому в С++ прототип звичайно пишеться так:

```
char fl();
```

С++ відрізняється від Сі способом задання порожнього списку параметрів. Якби попередній прототип мав місце в програмі Сі, то це б означало, що про параметри функції сказати нічого не можна. А в С++ це означає, що у функції немає параметрів. Тому в попередніх прикладах для позначення порожнього списку параметрів слово void не використовувалося. (Використання void для позначення порожнього списку параметрів не є помилковим, скоріше, воно є зайвим).

У С++ наступні два оголошення еквівалентні:

```
int f10 ; int f 1 (void);
```

2. У програмах С++ всі функції повинні мати прототипи. У Сі прототипи функцій рекомендуються, але технічно вони не обов'язкові, а в С++ прототипи необхідні.

3. Якщо в С++ функція має відмінний від void тип значення, що повертається, то інструкція return усередині цієї функції повинна містити значення даного типу.

4. У Сі, якщо тип значення, що повертає функцією, явно не заданий, функція за замовчуванням повертає значення цілого типу. У С++ такого правила немає. Отже, необхідно явно повідомляти тип значення всіх функцій, що повертає.

5. У програмах С++ можна вибирати місце для оголошення локальних змінних. У Сі локальні змінні можуть оголошуватися тільки на початку блоку, перед будь-якою інструкцією "дії". У С++ локальні змінні можуть оголошуватися в будь-якому місці програми. Одним з переваг такого підходу є те, що локальні змінні для запобігання небажаних побічних ефектів можна повідомляти поруч із місцем їхнього першого використання.

6. Для зберігання значень булевого типу (істина або неправда) у С++ визначений тип даних boot. У С++ також визначені ключові слова true й false - єдині значення, якими можуть бути дані типу bool. У С++ результатом виконання операторів відносини й логічні оператори є значення типу bool. У С++ у булевому виразі ненульове значення автоматично перетвориться в true, а нульове - в false, а також: true перетвориться в 1, а false в 0, якщо значення типу bool виявляється в цілому виразі. Додавання в С++ даних типу bool підсилює контроль типу й дає можливість розрізняти дані булевого й цілого типів. Природно, що використання даних булева типу не обов'язково, скоріше воно просто зручно.

У програмах Сі, при відсутності в командному рядку аргументів, функція main() звичайно оголошується так:

```
int main (void)
```

Однак у С++ таке використання слова void необов'язкове.

Ця коротка програма С++ не буде компілюватися, оскільки у функції sum() немає прототипу:

```
// Ця програма не буде компілюватися
// Эта программа не будет компилироваться
#include <iostream>
using namespace std;
void main()
{
    setlocale(LC_ALL, "Ukrainian");
    int a,b,c;
    cout << "Введите два числа: ";
    cin >> a >> b;
    c=sum(a, b);
    cout << "Сумма равна:" << c;
}
// Для цієї функції необхідний прототип
sum(int a, int b)
{
    return a+b;
```

```
}
```

Наступний приклад ілюструє той факт, що локальні змінні можна оголосити в будь-якому місці блоку:

```
#include <iostream>
using namespace std;
void main()
{
    setlocale(LC_ALL, "Ukrainian");
    int i; // локальна змінна, оголошена на
           //початку блока
    cout << "Введіть число:";
    cin >> i;
    // розрахунок факторіалу
    int j, fact=1; // зміні, оголошені перед
                   //інструкціями1
    for (j=i; j>=1; j--) fact=fact * j;
    cout << "Факторіал рівний: " << fact;
cin.get();
}
```

У наступній програмі створюється булева змінна outcome і їй присвоюється значення false. Потім ця змінна використовується в інструкції if.

```
#include <iostream>
using namespace std;
int main()
{
    bool outcome;
    outcome = false;
    if(outcome) cout << "істина"
    else cout << "хибне";
    return 0;
}
```

В результаті виконання програми на екрані з'являється слово хибне.

Введення в перевантаження функцій

Важливою й незвичайною можливістю C++ є перевантаження функцій (function overloading). Перевантаження функцій не тільки забезпечує механізм, за допомогою якого в C++ досягається один з типів поліморфізму, вона також формує те ядро, навколо якого розвивається все середовище програмування на C++.

У C++ дві або більше функції можуть мати те саме ім'я, відрізняючись або типом, або числом своїх аргументів, або й тим й іншим. Якщо дві або більше функції мають однакове ім'я, то кажуть, що вони перевантажені.

Перевантажені функції дозволяють спростити програми, допускаючи звертання до одного імені для виконання близьких за змістом дій.

Перевантажити функцію дуже легко: просто оголосити й визначите всі необхідні варіанти. Компілятор автоматично вибере правильний варіант виклику на підставі числа й/або типу використовуваних у функції аргументів.

Приклад !!!. Програма перевантаження одного імені для трьох типів даних:

```
#include <iostream>
using namespace std;
// Перевантаження abs() трьо способами
int abs(int n);
long abs(long n);
double abs(double n);
void main()
{
    setlocale(LC_ALL,"Ukrainian");
cout << "Абсолютна величина -10:" <<abs(-10)
<<endl;
    cout << "Абсолютна величина -10L:" << abs(-
10L) << endl;
    cout << "Абсолютна величина -10.01:" << abs(-
10.01) << endl;
cin.get();
}

// abs() для цілих
int abs(int n)
{
    cout << "В цілому abs()"<<endl;
    return n<0 ? -n : n;
}

// abs() для довгих цілих
long abs(long n)
{
    cout << "В длинном целом abs()"<<endl;
    return n<0 ? -n : n;
}

// abs() для дійсних подвійної точності
double abs(double n)
{
```

```
    cout << "В дійсному abs() подвійної
точності"<< endl;
    return n<0 ? -n : n;
      }
```

У програмі задано три функції abs(), своя для кожного типу даних. Усередині main() функція abs() викликається із трьома аргументами різних типів. Компілятор автоматично викликає правильну версію abs(), ґрунтуючись на використовуваному в аргументі типі даних. У результаті роботи програми на екран виводиться наступне:

У цілому abs ()

Абсолютна величина -10:10

У довгому цілому abs ( ) Абсолютна величина -10L: 10

У дійсному abs() подвійної точності Абсолютна величина -10.01: 10.01

Приклад. Перевантаження функції date() для одержання дати або у вигляді рядка, або у вигляді трьох цілих. В обох цих випадках функція виводить на екран передані їй дані.

```
#include <iostream>
using namespace std;
void date(char *date); // дата в виде строки
void date(int month, int day, int year); // дата в
виде чисел
int main()
{
      setlocale(LC_ALL,"Ukrainian");
    date("8/23/99");
    date( 8, 23, 99);
cin.get();
    return 0;
}
// Дата в виде строки
void date(char *date)
{
    cout << "Дата:" << date << "\n";
}
// Дата в виде целых
void date(int month, int day, int year)
{
    cout << "Дата:" << month << "/";
    cout << day << "/" << year << "\n";
}
```

Перевантаження функцій може забезпечити для функції більше зрозумілий інтерфейс. Оскільки дату дуже природно представляти або у вигляді рядка, або у вигляді трьох цілих чисел, що

містять місяць, день і рік, потрібно просто вибрати найбільш підходящу версію відповідно до ситуації.

Також перевантажені функції можуть також відрізнятися й числом аргументів, як показано в наведеному нижче прикладі:

```
#include <iostream>

using namespace std;
void f1(int a);
void f1(int a, int b);
void main()
{
    setlocale(LC_ALL,"Ukrainian");
    f1(10);
    f1(10, 20);
    cin.get();
}
void f1(int a)
{
    cout << "В f1(int a) \n";
}
void f1(int a, int b)
{
    cout << "В f1(int a, int b) \n";
}
```

У C++ підтримуються всі ключові слова C, а також ще 30 слів, які відносяться тільки до мови C++ (див. табл. 1.1).

Таблиця 1.1. Ключові слова C++

asm	const„cast	explicit	int	register	switch	union
auto	continue	extern	long	Reinterpret ^cast	template	unsigned
bool	default	false	mutable	return	this	using
break	delete	float	namespace	short	throw	virtual
case	do	for	new	signed	true	void
catch	double	friend	operator	sizepf	try	volatile
char	dynamic_ cast	goto	private	static	typedef	wcharj:
class	else	if	protected	static_cast	typeid	while
const	enum	inline	public	struct	typename	

Виведення у файловий потік

Для виведення у файловий потік необхідно оголосити об'єкт типу ofstream, вказавши ім'я необхідного вихідного файлу як символьний рядок, що показано нижче:

```
ofstream file_object("FILENAME.EXT");
```

Якщо вказати ім'я файлу при оголошенні об'єкта типу ofstream, C++, то створиться новий файл, або перезапишеться файл із таким же ім'ям, якщо він вже існує на диску.

Наступний прклад створює об'єкт типу ofstream і потім використовує оператор вставки для виведення декількох рядків тексту у файл TEST.DAT:

```
#include <iostream>
#include <fstream>
using namespace std;
void main()
{
      setlocale(LC_ALL,"Ukrainian");
    ofstream book_file("TEST.DAT");
   book_file << "Вчимося програмувати мовою C++, "
<< "Навіть щось виходить!" << endl;
   book_file << "KI рклить" << endl;
   book_file << "22.95" << endl;
}
```

Програма відкриває (створює) файл TEST.DAT і потім записує три рядки у файл, використовуючи оператор вставки

Вміст файлу:

Учимося програмувати мовою C++, Навіть щось виходить!

KI рулить!

$22.95

Читання із вхідного файлового потоку

Операції читання з файлу, використовуючи об'єкти типу ifstream можна вири ставши конструкцію:

```
ifstream input_file("filename.EXT");
```

Наступний приклад відкриває файл TEST.DAT, створений за допомогою попереднього прикладу, читає, а потім відображає перші три елементи файлу:

```
#include <iostream>
#include <fstream>
using namespace std;
void main()
{
    setlocale(LC_ALL,"Ukrainian");
    ifstream input_file("TEST.DAT") ;
```

```
    char one[64], two[64], three[64];
    input_file >> one;
    input_file >> two;
    input_file >> three;
    cout << one << endl;
    cout << two << endl;
    cout << three << endl;
    cin.get();
}
```

У результаті при запуску програми на екрані з'явиться наступна інформація:

```
учимося
програмувати
на
```

Читання цілого рядка файлового введення

Для читання цілого рядка із клавіатури застосовують команду cin.getline. Аналогічно можна читати об'єкти типу ifstream з файлу. Приклад читання всіх трьох рядків файлу TEST.DAT:

```
#include <iostream>
#include <fstream>
using namespace std;
void main()
{
   setlocale(LC_ALL,"Ukrainian");
   ifstream input_file("TEST.DAT");
   char one[64], two[64], three [64] ;
   input_file.getline(one, sizeof(one)) ;
   input_file.getline(two, sizeof(two));
   input_file.getline(three, sizeof(three)) ;
   cout << one << endl;
   cout << two << endl;
   cout << three << endl;
cin.get();
}
```

Програма успішно читає вміст файлу, тому що вона знає, що файл містить три рядки. Однак у багатьох випадках ваша програма не буде знати, скільки рядків місить файл. У таких випадках ваші програми будуть просто продовжувати читання вмісту файлу поки не зустрінуть кінець файлу.

Визначення кінця файлу

Звичайною файловою операцією є читання вмістів файлу, поки не зустрінеться кінець файлу. Щоб визначити кінець файлу, використовують функцію eof потокового об'єкта. Ця функція повертає

значення 0, якщо кінець файлу ще не зустрівся, і 1, якщо зустрівся кінець файлу. Використовуючи цикл while, ваші програми можуть безупинно читати вміст файлу, поки не знайдуть кінець файлу, як показано нижче:

```
while (! input_file.eof())
{ // Оператори }
```

У цьому випадку програма буде продовжувати виконувати цикл, поки функція eof повертає хибне (0). Наступний приклад використовує функцію eof для читання вмісту файлу TEST.DAT, поки не досягне кінця файлу:

```
#include <iostream>
#include <fstream>
using namespace std;
void main()
{
   setlocale(LC_ALL,"Ukrainian");
   ifstream input_file("TEST.DAT");
   char line[64];
   while (! input_file.eof())
   {
      input_file.getline(line, sizeof(line));
      cout << line << endl;
   }
cin.get();
}
```

Читання вмісту файлу по одному слову за один раз, поки не зустрінеться кінець файлу:

```
#include <iostream>
#include <fstream>
using namespace std;
void main()
{
   setlocale(LC_ALL,"Ukrainian");
   ifstream input_file("TEST.DAT");
   char word[64] ;
   while (! input_file.eof())
   {
      input_file >> word;
      cout << word << endl;   }
cin.get();
}
```

Читання вмісту файлу по одному символі за один раз, використовуючи функцію get, поки не зустріне кінець файлу:

```
#include <iostream>
```

```
#include <fstream>
using namespace std;
void main()
{
   setlocale(LC_ALL,"Ukrainian");
   ifstream input_file("TEST.DAT");
   char letter;
   while (! input_file.eof())
   {
       letter = input_file.get();
       cout << letter;
    }
cin.get();
}
```

Перевірка помилок при виконанні файлових операцій

Для відстежування помилок файлових операцій (перевірка існування файлу, перевірка успішності операції відкриття/ закриття/ запису), можна використати функцію fail файлового об'єкта. Якщо в процесі файлової операції помилок не було, функція поверне неправду (0). Однак, якщо зустрілася помилка, функція fail поверне істину. Наприклад, якщо програма відкриває файл, їй варто використати функцію fail, щоб визначити, чи відбулася помилка, як це показано нижче:

```
      ifstream input_file("FILENAME.DAT");
      if (input_file.fail())
      {
   cerr << "Помилка відкриття FILE.DAT" << endl;
      exit(1);
      }
```

Перевірка помилкових ситуацій файлових операцій подано у прикладі:

```
#include <iostream>
#include <fstream>
using namespace std;
void main()
{
   setlocale(LC_ALL,"Ukrainian");
    char line[256] ;
   ifstream input_file("TEST.DAT") ;
   if (input_file.fail()) cerr << "Помилка
відкриття TEST.DAT" << endl;
   else {
     while ((! input_file.eof()) && (!
                      input_file.fail()))
```

```
            {
        input_file.getline(line, sizeof(line));
        if (!input_file.fail())
                cout << line << endl;
             }
          }
cin.get();
}
```

Закриття файлу

При завершенні вашої програми операційна система закриє відкриті нею файли. Однак, як правило, якщо вашій програмі файл більше не потрібний, вона повинна його закрити. Для закриття файлу ваша програма повинна використати функцію close, як показано нижче:

```
input_file.close ();
```

Коли ви закриваєте файл, всі дані, які ваша програма писала в цей файл, скидаються на диск, і обновляється запис каталогу для цього файлу.

Керування відкриттям файлу

Для відкриття файлу в режимі до запису нелбхідно при його відкритті вказати другий параметр, як показано нижче:

```
ifstream output_file("FILENAME.EXT", ios::app);
```

Параметр `ios::app` вказує режим відкриття файлу. Режими відкриття подано у табл. 2.

Таблиця 2. Значення режимів відкриття.

Режим відкриття	Призначення
ios::app	Відкриває файл у режимі додавання, розташовуючи файловий покажчик наприкінці файлу.
ios::ate	Розташовує файловий покажчик наприкінці файлу.
ios::in	Указує відкрити файл для введення.
ios::nocreate	Якщо зазначений файл не існує, не створювати файл і повернути помилку.
ios::noreplace	Якщо файл існує, операція відкриття повинна бути перервана й повинна повернути помилку.
ios::out	Указує відкрити файл для виводу.
ios::trunc	Скидає (перезаписує) містимо, з існуючого файлу.

Операція відкриття файлу для запису, використовуючи режим `ios::noreplace`, запобігає перезапис існуючого файлу:

```
ifstream output_file("FIlename.EXT",
  ios::out | ios::noreplace);
```

Виконання операцій читання й записи

Для читання та запису масивів і структур можна використати функції read й write з вказанням буферу даних, у який дані будуть читатися або з якого вони будуть записуватися, а також довжину буфера в байтах, як показано нижче:

```
input_file.read(buffer, sizeof(buffer)) ;
 output_file.write(buffer, sizeof(buffer));
```

Приклад використання функції write для виводу вмісту структури у файл EMPLOYEE.DAT:

```
#include <iostream>
#include <fstream>
using namespace std;
void main()
{
   setlocale(LC_ALL,"Ukrainian");
   struct employee
  {
   char name[64];
   int age;
   float salary;
   } worker = { "Джон Дой", 33, 10000.0 };
   ofstream emp_file("EMPLOYEE.DAT") ;
   emp_file.write((char *) &worker,
sizeof(employee));
cin.get();
}
```

Функція write одержує покажчик на символьний рядок. Символи (char *) являють собою оператор приведення типів, що інформує компілятор, що ви передаєте покажчик на інший тип.

Наступний приклад використовує метод read для читання з файлу інформації про службовця:

```
#include <fstream>
using namespace std;
void main()
{
   setlocale(LC_ALL,"Ukrainian");
   struct employee
   {
      char name [64] ;
      int age;
      float salary;
   } worker = { "Джон Дой", 33, 25000.0 };
   ifstream emp_file("EMPLOYEE.DAT");
   emp_file.read((char *) &worker,
```

```
                sizeof(employee));
   cout << worker.name << endl;
   cout << worker.age << endl;
   cout << worker.salary << endl;
cin.get();
}
```

ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ №1

Перший рівень

Завдання 1. Розробити програму на мові С++, що реалізує запис у файл інформації про студента (10 позицій) у два способи: як текстова інформація та у вигляді структури.

Другий рівень

Завдання 1. Розробити програму на мові С++, що реалізує:

1.1 функцію запису у типізований файл INFO.DAT згідно варіанту;

1.2. функцію читання з файлу INFO.DAT згідно варіанту.

Варіанти завдань для текстових:

2. Сформувати файл, що містить прізвища, стать, рік народження та групу крові N донорів. Використовуючи сформований файл, надрукувати прізвища донорів жіночої статі, що мають IV групу крові.

2. Сформувати файл, що містить прізвища, стать, групу крові та резус-фактори N донорів. Використовуючи сформований файл, надрукувати прізвища донорів, що мають I групу крові та резус-фактор.

2. Сформувати файл, що містить дані про N книг з програмування вашої особистої бібліотеки (прізвище автора та його ініціали, назва книги, назва видавництва). Використовуючи сформований файл, надрукувати прізвища авторів та назви книг видавництва "Наука".

3. Сформувати файл, що містить дані про N книг по ЕОМ вашої особистої бібліотеки (прізвище автора та його ініціали, назва книги, назва видавництва). Використовуючи сформований файл, надрукувати прізвища авторів та назви книг, що видані не пізніше 1988 р.

4. Сформувати файл, що містить інформацію про N побутових магнітофонів (марка, виробник, ціна). Використовуючи сформований файл, надрукувати інформацію про магнітофони, які виготовлені в Україні і з ціною у заданих межах (межі вибираються самостійно).

5. Сформувати файл, що містить інформацію про N поїздів, які відправляються з вокзалу станції м. Хмельницький (номер поїзда, станція призначення, час відправлення, час в дорозі). Використовуючи сформований файл, надрукувати інформацію про поїзди, що відправляються не пізніше 21 години.

6. Сформувати файл, що містить інформацію про N поїздів, які відправляються з вокзалу станції м. Хмельницький (номер поїзда, станція

призначення, час відправлення, час прибуття). Використовуючи сформований файл, надрукувати інформацію про поїзди, час в дорозі яких не перевищує 17 годин.

7. Сформувати файл, що містить інформацію про N телевізорів (марка, ціна, виробник). Використовуючи сформованний файл, надрукувати інформацію про найдешевший телевізор.

8. Сформувати файл, що містить прізвища N студентів та оцінки кожного студента за результатами п'яти іспитів. Використовуючи сформований файл, надрукувати прізвища студентів, які мають найвищий бал.

9. Сформувати файл, що містить прізвища N студентів та оцінки кожного студента за результатами п'яти іспитів. Використовуючи сформований файл, надрукувати прізвища студентів, що здали всі іспити на "5".

10. Сформувати файл-протокол лижних гонок, що містить прізвища N учасників, час старту, час фінішу для кожного учасника (години, хвилини, секунди). Використовуючи сформований файл, надрукувати прізвище учасника, що посів перше місце.

11. Сформувати файл-довідник, що містить прізвища N співробітників відділу та номери їхніх домашніх телефонів (інформація про кожного співробітника вводиться в одну літерну змінну в літерному вигляді у наступному порядку: прізвище - п'ятизначне число, що визначає номер телефону, наприклад, Петров - 66803). Використовуючи сформований файл, роздрукувати номер телефону даного співробітника (пошук номера телефону за прізвищем) у звичайному вигляді (наприклад, 6-68-03).

12. Сформувати файл, що містить інформацію про N студентів (прізвище, стать, рік та місяць народження). Використовуючи сформований файл, надрукувати прізвища студентів, які народилися влітку.

13. Сформувати файл, що містить інформацію про N студентів (прізвище, стать, рік народження). Використовуючи сформований файл, надрукувати прізвища студентів жіночої статі з вказівкою на вік.

14. Сформувати файл, що містить прізвища N студентів групи (у довільному порядку). Використовуючи сформований файл, надрукувати прізвища студентів в алфавітному порядку.

15. Сформувати файл, що містить інформацію про дати народження N ваших друзів (інформація про одного друга вводиться в одну літерну змінну у наступному порядку: прізвище-ДД.ММ.РРРР, наприклад, Сидоров - 7.03.1980). Використовуючи сформований файл, надрукувати прізвища друзів, що народилися восени.

16. Сформувати файл, що містить інформацію про N людей, які мають автомобілі (прізвище власника, марка автомобіля, колір).

Використовуючи сформований файл, надрукувати прізвища тих, хто має "Ладу" червоного кольору.

17. Сформувати файл, що містить інформацію про N людей, що мають автомобілі (прізвище власника, марка автомобіля, його повний номер, наприклад, В34-61ХМ). Використовуючи сформований файл, надрукувати прізвища тих, у кого номер автомобіля містить цифри 72-15 (якщо таких немає, вивести відповідне повідомлення).

18. Сформувати файл, що містить інформацію про N магазинів вашого міста: назва магазину (універмаг, "Продукти" і т. д.), його номер та
адреса (вулиця). Використовуючи сформований файл, надрукувати інформацію про ті, що знаходяться на вулиці Кам'янецькій.

19. Сформувати файл, що містить інформацію про всі технікуми вашого міста: назва, адреса (інформація про кожного співробітника вводиться в одну літерну змінну в літерному вигляді у наступному порядку: назва вулиці, номер будинку, наприклад, комерційний, Кам'янецька, 114). Використовуючи сформований файл, визначити, чи є у вашому місті медичний технікум, та, якщо є, надрукувати його адресу у звичайному вигляді (наприклад, вул. Кам'янецька, 114).

20. Сформувати файл, що містить інформацію про N магазинів вашого міста: назва магазину (універмаг, "Продукти" і т. д.), його номер та адреса (вулиця). Використовуючи сформований файл, надрукувати інформацію про всі магазини.

21. Сформувати файл, що містить інформацію про N співробітників відділу: прізвище, ім'я, по батькові, посада (інженер, бухгалтер, програміст і т. д), оклад. Використовуючи сформований файл, підрахувати, скільки є програмістів у відділі, надрукувати їхні прізвища, імена, по батькові та оклади.

22. Сформувати файл, що містить інформацію про N співробітників відділу: прізвище, ім'я, по батькові, посада, оклад. Використовуючи сформований файл, надрукувати прізвища, імена, по батькові, посади співробітників, що мають найменший оклад.

23. Сформувати файл, що містить інформацію про N співробітників відділу: прізвище, ім'я, по батькові, посада, оклад. Використовуючи сформований файл, визначити та надрукувати середній оклад по відділу, а також інформацію про співробітника, що має оклад, найближчий до середнього.

24. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Використовуючи сформований файл, визначити та надрукувати прізвища студентів 1-го курсу, що отримують стипендію.

25. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети).

Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що здали сесію без '^трійок" (оцінки вказати).

26. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що здали сесію з однією "трійкою".

27. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що здали сесію на "відмінно" (вказати курс і групу).

28. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, курс, група, адреса, телефон). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що живуть у гуртожитку.

29. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, адреса, телефон). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що мають телефон. Вказати номер.

Третій рівень

Завдання 1. Розробити програму на мові С++, що реалізує:
1.1 функцію запису RW() у типізований файл INFO.DAT у вигляді структури згідно варіанту;
1.2 функцію читання RD() з файлу INFO.DAT для виведення вмісту файлу на екран;
1.3 функцію запису RW() у файл INFO.TXT текстової інформації згідно варіанту;
1.4. функцію читання RD() з файлу INFO.TXT для виведення вмісту файлу на екран;

Означені функції RW() та RD() повинні в процесі їх використання головній програмі **перевантажуватися.**

Варінти завдань для типізованого файлу INFO.DAT:

1. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, адреса, телефон). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що живуть на заданій вулиці.
2. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що народилися до заданої' дати (вказати дату народження).
3. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, стать, курс, група). Використовуючи сформований

файл, визначити та надрукувати прізвища студентів чоловічої статі (вказати курс і групу).

4. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що народились у заданому році.

5. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що народились під знаком Рака (22.06 - 21.07).

6. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що народились взимку.

7. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, стать, курс, група). Використовуючи сформований файл, визначити та надрукувати прізвища студентів жіночої статі, вказавши курс і групу.

8. Сформувати файл, що містить інформацію гро N студентів (прізвище та ім'я, курс, група, адреса, телефон). Використовуючи сформований файл, визначити та надрукувати прізвища студентів, що не мають телефону (вказати адресу).

9. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Використовуючи сформований файл, визначити та надрукувати середній бал по кожному іспиту.

10. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, стать, курс, група). Впорядкувати інформацію та вивести її в алфавітному порядку за прізвищами, вказавши курс та групу.

11. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, адреса, телефон). Впорядкувати інформацію та вивести її в алфавітному порядку за адресами, вказавши прізвище та адресу.

12. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Впорядкувати інформацію та вивести її у порядку старшинства за віком з вказанням прізвища і дати народження.

13. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, адреса, телефон). Впорядкувати інформацію та вивести її у порядку зростання номерів телефону з вказанням номера телефону та прізвища.

14. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її за успішністю з всіх предметів, вказавши середній бал екзаменаційних оцінок.

15. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, дата народження, стать, адреса, телефон, курс, група,

результати сесії (4 предмети). Впорядкувати інформацію та вивести її по курсах, вказавши курс та прізвище студента даного курсу.

16. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Впорядкувати інформацію та вивести її по днях народження в даному місяці з вказанням дати.

17. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її по успішності для кожного курсу, вказавши екзаменаційні оцінки.

18. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її за кількістю зданих на "відмінно" іспитів для кожної групи.

19. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її за кількістю зданих на "відмінно" іспитів для кожного курсу.

20. Сформувати файл, що містить інформацію про N поїздів, які відправляються з вокзалу станції м. Хмельницький (номер поїзда, станція призначення, час відправлення, час в дорозі). Використовуючи сформований файл, надрукувати інформацію про поїзди, що відправляються не пізніше 18 години.

Варінти завдань текстового файлу INFO.DAT

1. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, стать, курс, група). Впорядкувати інформацію та вивести її в алфавітному порядку за прізвищами, вказавши курс та групу.

2. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, адреса, телефон). Впорядкувати інформацію та вивести її в алфавітному порядку за адресами, вказавши прізвище та адресу.

3. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Впорядкувати інформацію та вивести її у порядку старшинства за віком з вказанням прізвища і дати народження.

4. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, адреса, телефон). Впорядкувати інформацію та вивести її у порядку зростання номерів телефону з вказанням номера телефону та прізвища.

5. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її за успішністю з всіх предметів, вказавши середній бал екзаменаційних оцінок.

6. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, дата народження, стать, адреса, телефон, курс, група,

результати сесії (4 предмети). Впорядкувати інформацію та вивести її по курсах, вказавши курс та прізвище студента даного курсу.

7. Сформувати файл, що містить інформацію про N студентів (прізвище та ім'я, дата народження). Впорядкувати інформацію та вивести її по днях народження в даному місяці з вказанням дати.

8. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її по успішності для кожного курсу, вказавши екзаменаційні оцінки.

9. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її за кількістю зданих на "відмінно" іспитів для кожної групи.

10. Сформувати файл, що містить інформацію про N студентів: прізвище та ім'я, курс, група, результати сесії (4 предмети). Впорядкувати інформацію та вивести її за кількістю зданих на "відмінно" іспитів для кожного курсу.

11. Сформувати файл, що містить інформацію про N поїздів, які відправляються з вокзалу станції м. Хмельницький (номер поїзда, станція призначення, час відправлення, час в дорозі). Використовуючи сформований файл, надрукувати інформацію про поїзди, що відправляються не пізніше 18 години.

12. Сформувати файл, що містить інформацію про N поїздів, що відправляються з вокзалу станції м. Хмельницький (номер поїзда, станція призначення, час відправлення, час прибуття). Використовуючи сформований файл, надрукувати інформацію про поїзди, час в дорозі яких не перевищує 10 годин.

13. Сформувати файл, що містить інформацію про N телевізорів (марка телевізора, його ціна, виробник). Використовуючи сформованний файл, надрукувати інформацію про найдорожчий телевізор.

14. Сформувати файл, що містить прізвища N студентів та оцінки кожного студента за результатами п'яти іспитів Використовуючи сформований файл, надрукувати прізвища студентів, що мають найнижчий бал.

15. Сформувати файл, що містить прізвища N студентів та оцінки кожного студента за результатами п'яти іспитів. Використовуючи сформований файл, надрукувати прізвища студентів, що здали всі іспити на "чотири".

16. Сформувати файл-протокол лижних гонок, що містить прізвища N учасників, час старту і фінішу для кожного учасника (години, хвилини, секунди). Використовуючи сформований файл, надрукувати прізвище учасника, що посів третє місце.

17. Сформувати файл, що містить інформацію про N людей, які мають автомобілі (прізвище власника, марка автомобіля, його номер). Номер записується повністю, наприклад В34-61ХМ. Використовуючи

сформований файл, надрукувати прізвища тих, Чий номер містить цифри 12-13 (якщо таких немає, вивести відповідне повідомлення).

18. Сформувати файл, що містить інформацію про N магазинів вашого міста: назва магазину (універмаг, "Продукти" і т. д.), його номер та адреса (вулиця). Використовуючи сформований файл, надрукувати інформацію про ті, що знаходяться на вулиці Університетській.

19. Сформувати файл, що містить інформацію про всі технікуми вашого міста: назва, адреса (інформація про кожного співробітника вводиться в одну літерну змінну в літерному вигляді у наступному порядку: назва вулиці, номер будинку, наприклад, комерційний, Кам'янецька, 114). Використовуючи сформований файл, визначити, чи є у вашому місті юридичний технікум, та, якщо є, надрукувати його адресу у звичайному вигляді (наприклад, вул. Кам'янецька, 114).

20. Сформувати файл, що містить інформацію про N магазинів вашого міста: назва магазину (універмаг, "Продукти" і т. д.), його номер та адреса (вулиця). Використовуючи сформований файл, надрукувати інформацію про всі продуктові магазини.

Додаток А – Приклад титульного листа звіту лабораторних робіт

Міністерство освіти, науки, молоді та спорту України
Хмельницький національний університет
Кафедра системного програмування

Звіт
про виконання лабораторної роботи №…
з навчальної дисципліни
«Об'єктно-орієнтоване програмування»
на тему «…..»

Виконав: студент групи (*шифр групи*)
(*П.І.Б. студента*)

Перевірив: (*П.І.Б. викладача*)

Хмельницький (*рік*)