**Міністерство освіти і науки України Національний університет «Львівська політехніки»**

**Кафедра ЕОМ**



**Курсова робота**

з дисципліни «Програмування, частина 2 (Об'єктно-орієнтоване програмування)»

на тему:

**«Базові принципи об’єктно-орієнтованого програмування»**

Індивідуальне завдання: «Бібліотека»

**Виконав:**

ст.гр. КІ-15

Блищак Р.С.

**Перевірив**:

Козак Н.Б.

Львів -2020

**Анотація**

Метою курсової роботи є створення спрощеної моделі функціонування бібліотеки. Написання програмного забезпечення було виконано мовою програмування C++ із використанням об’єктно-орієнтованого підходу для формування програми у вигляді набору сутностей та об’єктів, що взаємодіють. Програма включає у собі можливість роботи із текстовими файлами, а саме: запис та зчитування даних, пошук необхідної інформації та її видалення, а також сортування за певною характеристикою для зручнішого використання та керування інформацією. Програмне забезпечення передбачає кілька рівнів доступу, що дозволяє встановити роль користувача та надати йому лише ті можливості керування інформацією, що необхідні конкретно даній особі. Інтерфейс програми написано українською мовою.

Зміст

[Вступ 6](#_Toc10500182)

[1.Визначення поняття ООП, його основні принципи та обгрунтування його технологій. 7](#_Toc10500183)

[1.1ООП як основа сучасного програмування 7](#_Toc10500184)

[1.2 Визначення ООП 8](#_Toc10500185)

[1.3 Фундаментальні поняття ООП 9](#_Toc10500186)

[1.4 Переваги обраного методу. 10](#_Toc10500187)

[2 Аналіз та розробка алгоритму згідно індивідуального завдання 11](#_Toc10500188)

[2.1 Опис алгоритму ( діаграма прецедентів ) 11](#_Toc10500189)

[2.2 Аналіз особливостей алгоритму. 13](#_Toc10500190)

[2.3 Опис алгоритму у псевдокоді 14](#_Toc10500191)

[2.4.Блок-схема алгоритму (для класу Читача і Адміністратора): 15](#_Toc10500192)

[3. Програмна реалізація 17](#_Toc10500193)

[3.1.Опис послідовної версії програмної реалізації 17](#_Toc10500194)

[3.1.1 Опис структур даних програми 17](#_Toc10500195)

[3.1.2 Опис реалізації основних функій/методів 18](#_Toc10500196)

[3.2 Опис програмної реалізації алгоритму 20](#_Toc10500197)

[3.2.1 Діаграма класів 20](#_Toc10500198)

[3.2.2. Схема викликів функції 22](#_Toc10500199)

[4 Відлагодження та тестування реалізованої програми згідно індивідуального завдання 22](#_Toc10500200)

[4.1 Опис структури програми 22](#_Toc10500201)

[4.2Відлагодження та тестування окремих частин програми 22](#_Toc10500202)

[4.3 Відлагодження та тестування програми в цілому 23](#_Toc10500203)

[5.Відлагодження та тестування реалізованої програми згідно індивідуального завдання 24](#_Toc10500204)

[5.1. Аналіз структурної складності виконання програми 24](#_Toc10500205)

[5.2. Аналіз часу виконання програми в залежності від об'єму вхідних даних 26](#_Toc10500206)

[Висновок 30](#_Toc10500207)

[Використана література: 30](#_Toc10500208)

**Завдання на курсову роботу**

Згідно із отриманим індивідуальним завданням метою курсової роботи є реалізація мовою C++ із застосуванням об’єктно-орієнтованого підходу предметної галузі “Бібліотека”. Програма повинна підтримувати можливості роботи з файлами : запис, зчитування, пошук, видалення та оновлення даних, а також їх сортування за певним критерієм. Для типу користувачів “Клієнт” передбачена система пунктів для кращої орієнтації у програмі. Для користувачів, що мають права адміністратора повинні бути реалізовані методи, що дозволяють керувати даними.

# Вступ

**Об’єктно-орієнтоване програмування** (ООП) — одна з парадигм програмування, яка розглядає програму як множину об'єктів, що взаємодіють між собою. В основі ООП покладено три основні положення: інкапсуляцію, успадкування та поліморфізм. Однією з переваг ООП є краща модульність програмного забезпечення, що дозволяє групам розробників працювати незалежно один від одного, розробляючи окремі функціональні компоненти програми . Попри те, що парадигма ООП з'явилась в 1960-тих роках, вона не мала широкого застосування до 1990-тих, коли розвиток комп'ютерів та комп'ютерних мереж дозволив писати надзвичайно об'ємне і складне програмне забезпечення, що змусило переглянути підходи до написання програм та покращити їх модульність. Однією з таких мов програмування стала мова С++. У 1990-х р. вона стала однією із найпопулярніших мов програмування загального призначення. Розробником цієї мови став професор Техаського університету Б’єрн Страуструп. На початку свого існування ця мова дістала назву «Сі з класами». Пізніше у 1983 році Страуструп перейменував мову на C++. У 1998 році було ратифіковано міжнародний стандарт мови С++: ISO/IEC 14882 «Standard for the C++ Programming Language». Сьогодні цю мову використовують для розробки програмного забезпечення, написання драйверів, системного програмування, потужних серверних та клієнтських програм. Мова C++ мала значний вплив на інші мови програмування такі як Java, C# та ін.

**Метою виконання** курсової роботи є засвоєння основних методів та алгоритмів об’єктно-орієнтованого програмування мови С++ та їх практичне застосування у описі сутностей конкретної сфери.

# Визначення поняття ООП, його основні принципи та обгрунтування його технологій.

## 1.1ООП як основа сучасного програмування

Раніше програми розроблялися із застосуванням процедурного підходу, що полягав у поділі проекту на окремі більш дрібні складові – функції. Із розростанням проектів та ускладненням завдань з’явилась необхідність у більш раціональному підході у написанні програмного забезпечення та ефективному поділі поставленого завдання таким чином, щоб не виникало проблем при розробці окремих складових проекту. Для подолання цієї проблеми було розроблено структурне програмування. Структурне програмування, на відміну від свого попередника — стихійного програмування, має певні правила та засоби, які суттєво вдосконалили процес написання програм. Однак через обмеженість структурного програмування дуже важко писати програми ще більшої складності, яких вимагає сучасний ринок праці. Принципово новим підходом стало впровадження об’єктно-орієнтованого програмування. Першою мовою програмування, в якій були запропоновані базові поняття, які згодом об’єдналися у парадигму, була Симула, але термін “об'єктна орієнтованість” не можна було застосувати в контексті використання цієї мови. У 1967 році були запропоновані революційні ідеї: об'єкти, класи, віртуальні методи і ін., проте це не було сприйнято сучасниками як щось революційне та здатне змінити підхід до розробки програмного забезпечення. Погляд на програмування «під іншим кутом» (відмінним від процедурного) запропонували Алан Кей і Ден Інгаллс в мові Smalltalk. Тут поняття класу стало основотвірною ідеєю для всіх інших конструкцій мови (тобто клас в Смолток є примітивом, за допомогою якого описані більш складні конструкції). Саме ця мова стала першою широко поширеною об'єктно- орієнтованою мовою програмування. Станом на сьогодні кількість прикладних мов програмування (список мов), що реалізують об'єктно-орієнтовану парадигму, є найбільшою порівняно з іншими парадигмами.

Отже, впровадження ООП стало вагомим внеском у розвитку сучасного програмування раціональним шляхом мінімізації коду та покращенням ефективності.

## 1.2 Визначення ООП

*Об'єктно-орієнтоване програмування (ООП) - методологія програмування, заснована на представленні програми у вигляді сукупності об'єктів, кожен з яких є екземпляром певного класу, а класи можуть утворюють ієрархію спадкування*. **[2]** Лука Карделла і Мартін Абаді побудували теоретичне обгрунтування ООП і класифікацію на основі цього обгрунтування. Вони відзначають, що виділені ними поняття і категорії разом зустрічаються далеко не у всіх об’єктно-орієнтованих мовах, більшість мов підтримують лише підмножини теорії, а часом і своєрідні відхилення від неї.

Ключові поняття:

* квантифікація змінних типу (універсальна, екзистенційна, обмежена);
* підтипізація (англ. subtyping - відносини «супертип-підтип»);
* включення (англ. subsumption) - окремий випадок підтипізації;
* об'єкт;
* об'єктний тип (фактично, спеціальна форма «запису», в число полів якої входять функції першого класу);
* клас.

*Клас – це сутність, що описує абстрактну поведінку. Об'єктні типи будуються на основі класу за допомогою додавання різних приватних полів і методів. Об'єкт (тобто значення об'єктного типу, традиційно називається «екземпляром класу») породжується конструктором на основі початкових параметрів.* **[2]**

## 1.3 Фундаментальні поняття ООП

До фундаментальних понять (принципів), використаних у переважній більшості визначень об'єктно-орієнтованого програмування належать:

* *Абстракція даних – виділення лише значущої інформації і виключення її з розгляду незначною. В ООП розглядають лише абстракцію даних (нерідко називаючи її просто «абстракцією»), маючи на увазі набір значущих характеристик об'єкта, доступних решті частини програми.*
* *Інкапсуляція – властивість системи, що дозволяє об'єднати дані і методи, що працюють з ними, в класі. Одні мови (наприклад, С ++, Java або Ruby) ототожнюють інкапсуляцію з приховуванням, але інші (Smalltalk, Eiffel, OCaml) розрізняють ці поняття.*
* *Спадкування – властивість, що дозволяє описати новий клас на основі вже існуючого, що передбачає часткове або абсолютне отримання властивостей чи методів батьківського класу. Клас, від якого відбувається спадкування, називається базовим, батьківським або суперкласом. Новий клас – нащадком, спадкоємцем, дочірнім або похідним класом.*
* *Поліморфізм підтипів – властивість, що дозволяє використовувати об'єкти з однаковим інтерфейсом без інформації про тип і внутрішню структуру об'єкта. Інший вид поліморфізму – параметричний.*
* *Клас – абстрактні характеристики деякої сутності, включаючи характеристики самої сутності (її атрибути або властивості) та дії, які вона здатна виконувати (її поведінки, методи або можливості). Класи вносять модульність та структурованість в об'єктно-орієнтовану програму. Як правило, клас має бути зрозумілим для осіб, що не мають відношення до програмування, проте знаються на предметній області, що, у свою чергу, значить, що клас повинен мати значення в контексті. Також, код реалізації класу має бути досить самодостатнім. Властивості та методи класу, разом називаються членами даного класу.*
* *Об’єкт – сутність в адресному просторі обчислювальної системи, що з'являється при створенні екземпляра класу (наприклад, після запуску результатів компіляції і зв'язування вихідного коду на виконання).* **[1]**

## 1.4 Переваги обраного методу.

Оптимальність та раціональність вибору того чи іншого методу є однією із фундаментальних основ при розробленні програмного забезпечення для розв’язання конкретної задачі. Від вибору підходу при створенні програми залежить час виконання програми, ресурсозатратність і т.д. Тому грамотне та доцільне рішення може суттєво спростити процес розроблення та вдосконалення програмного забезпечення в майбутньому. Чим більшою стає програма, тим гостріше постає проблема про важливість чітко окресленого поділу на компоненти, абстрагуючись від незначних деталей і в той же час проектуючи складові так, щоб вони могли описати сутності, на базі яких буде передбачено створення об’єктів та описано правила їх взаємодії. Однією із основних переваг використання саме об’єктно-орієнтованого підходу є можливість створювати розширювані та модульні системи, що й дає перевагу над традиційними методами програмування. Розширюваність передбачає, що існуючу систему можна доповнити новими компонентами, причому без внесення до неї будь-яких змін. Компоненти можуть бути додані на етапі виконання.

# 2 Аналіз та розробка алгоритму згідно індивідуального завдання

## 2.1 Опис алгоритму ( діаграма прецедентів )

Моїм завданням було програмно продемонструвати роботу Бібліотеки.

У моїй програмі було реалізовано можливості керування бібліотекою, а саме:

* Створення книг окремих жанрів, заповнення інформації про книги, наявності в бібліотеці, інших характеристик.
* Керування працівниками бібліотеки, заповнення інформації про працівників.

**Працівник**

Персонал бібліотеки несе відповідальність за правильне функціонування усіх його частин. За допомогою персоналу відбувається взаємодія між читачем та книгами. Працівники можуть додавати нові книжки, виконувати інші важливі з точки зору їх функціональності у бібліотеці функції.

**Книги**

Програмно сутність «книга» повинна бути максимально наближено передати її роль у реальному світі.

Книга містить назву, автора, опис, рік видання та ще низку властивостей.

Я вважаю, що обраних властивостей книги буде достатньо, щоб приблизити книгу до реальності.

**Бібліотека**

Працівник

Книга

**Рис.1.Діаграма прецедентів програми**

Актори:

**Книга:**

Це об’єкт , що містить такі характеристики:

* Назва книги
* Автор книги
* Назва видавництва
* Рік видання
* Короткий опис книги
* Кількість сторінок книги
* Наявність книги в бібліотеці

**Працівник**:

* Додавання нових книг;
* ПІБ працівника;
* Контактний номер телефону;
* Перегляд списку працівників.

Як видно на діаграмі, адміністратор(вид персоналу «Працівник») та книга не мають спільних властивостей.

## 2.2 Аналіз особливостей алгоритму.

Розглянемо детальніше кожну функцію нашої діаграми.

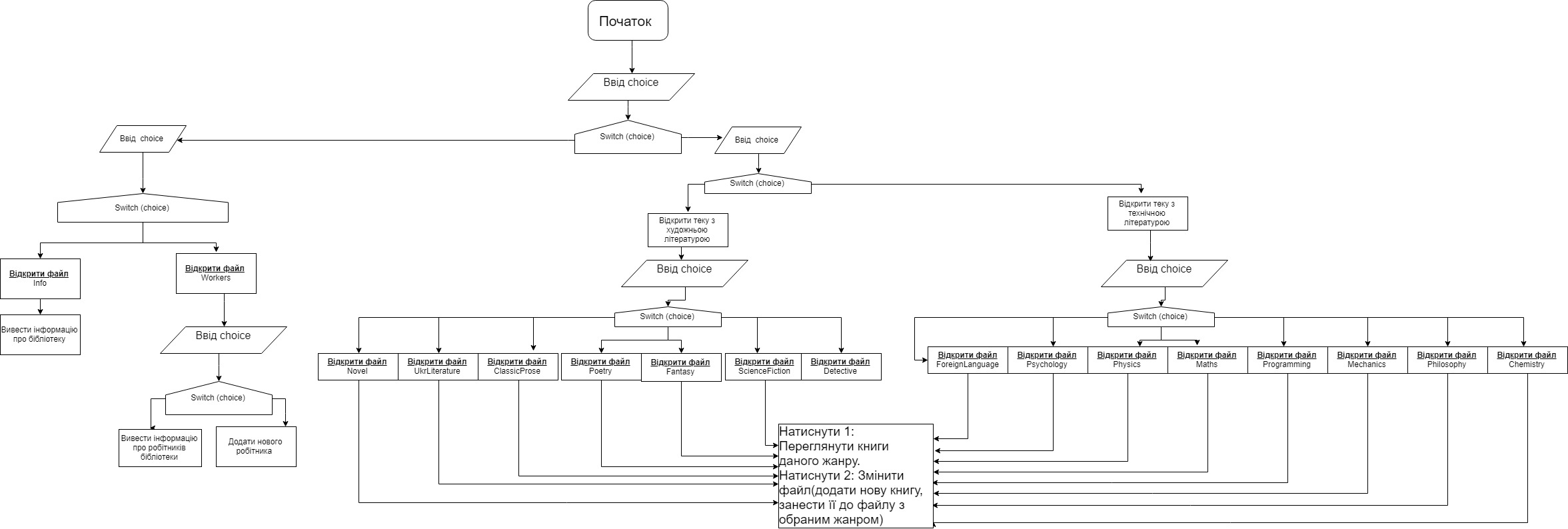
1. Властивість «Назва » - дозволяє користувачу дізнатися, що за книжка.
2. Властивість «Автор» - дозволяє користувачу дізнатися хто ж написав цю книжку.
3. Властивість «Назва видавництва » - дозволяє користувачу дізнатися хто надрукував цю книжку.
4. Властивість «Рік» - дозволяє користувачу дізнатися коли видали цю книжку.
5. Властивість «Опис» - дозволяє користувачу дізнатися про що ця книжка.
6. Властивість «Кількість сторінок» - дозволяє користувачу дізнатися який об’єм цієї книжки.
7. Властивість «Наявність» - дозволяє користувачу дізнатися чи є ця книжка у бібліотці.
8. Функція «Додати книгу» - дозволяє збільшити кількість книг у бібліотеці, а також додати нову книжку тим самим збільшити різновид книг.
9. Властивість «ПІБ» - дозволяє дізнатись хто працює в бібліотеці.
10. Властивість «Контактний номер» - дозволяє звязатись із працівником бібліотеки.
11. Функція «Перегляд персоналу» - дозволяє дізнатись хто саме працює у бібліотец

## 2.3 Опис алгоритму у псевдокоді

Основними задачами програми є:

* + Створювати файл для окремих жанрів книг
  + Вносити дані про книги
  + Виводити внесені дані
  + Створювати файл для працівників
  + Вносити дані про працівників
  + Багатократна перевірка на правильність введених даних
  + Виводити внесені дані про працівників
  + Вихід з програми

## 2.4.Блок-схема алгоритму (для класу Читача і Адміністратора):



Пояснення до блок схеми:

Меню доступу:

1. Відкривається меню, де користувач вибирає можливості.
2. Якщо користувач вводить 1, то він переходить до вкладки інформація.
3. Якщо користувач вводить 2, то він переходить до вкладки перегляд книг.
4. Якщо користувач вводить 3, то він завершує роботу.

Вкладка інформація:

1. Якщо користувач вводить 1, то у нього з’являється інформація про бібліотеку;
2. Якщо користувач вводить 2, то у нього з’являється інформація про працівників;
3. Якщо користувач вводить 3, то у нього з’являється можливість вернутися назад до головного меню.

Вкладка книги:

1. Якщо користувач вводить 1, то у нього з’являється можливість перейти до художньої літератури;
2. Якщо користувач вводить 2, то у нього з’являється можливість до технічної літератури;
3. Якщо користувач вводить 3, то у нього з’являється можливість вернутися назад до головного меню.

Художня література:

Ця література ділиться на:

1. Новели;
2. Українську літературу;
3. Класичну прозу;
4. Поезії;
5. Фантастику;
6. Наукову;
7. Детективи.

Вибравши одну із цих категорій користувачу відкриваються можливості:

1. Якщо користувач вводить 1, то у нього з’являється можливість дізнатися інформацію про ці книги;
2. Якщо користувач вводить 2, то у нього з’являється можливість додати нову книгу.

Технічна література:

Ця література ділиться на:

1. Психологію;
2. Фізику;
3. Математитичку;
4. Програмування;
5. Іноземну мову;
6. Хімію;
7. Філософію;
8. Механіку.

Вибравши одну із цих категорій користувачу відкриваються можливості:

1. Якщо користувач вводить 1, то у нього з’являється можливість дізнатися інформацію про ці книги;
2. Якщо користувач вводить 2, то у нього з’являється можливість додати нову книгу.

# 3. Програмна реалізація

## 3.1.Опис послідовної версії програмної реалізації

### 3.1.1 Опис структур даних програми

Програма створює текстові документи в залежності від вибору користувача. Їх може бути 16: «Classic Fiction.txt», «Poetry.txt», «Fantasy.txt», «Detective.txt», «Novel.txt», «Science Fiction.txt», «Ukrainian Literature.txt», «Psychology.txt», «Physics.txt», «Maths.txt», «Programming.txt», «Foreign Languages.txt», «Chemistry.txt», «Philosophy.txt», «Mechanics.txt», «WorkerInfo.txt»

Текстові файли від «Classic Fiction.txt» до «Mechanics.txt»

Записуються відомості про книгу, такі як:

* Назва книги
* Автор книги
* Назва видавництва
* Рік видання
* Короткий опис книги
* Кількість сторінок книги
* Наявність книги в бібліотец.

Текстовий файл «WorkerInfo.txt»

Записуються відомості про працівника, такі як:

* ПІБ працівника;
* Контактний номер телефону;
* Перегляд працівників.

В програмі наявні змінні:

* Цілого типу для виконання умови switch;
* Цілого типу для зчитування з консолі, виконання обчислень та виведення в консоль**/**файл;
* Типу string для зчитування з консолі та виведення в консоль**/**файл;
* Об’єкти класів для виконання методів.

### 3.1.2 Опис реалізації основних функій/методів

Функція «GetInf», використовуючи файловий ввід <ifstream>, зчитує з файлів «Classic Fiction.txt», «Poetry.txt», «Fantasy.txt», «Detective.txt», «Novel.txt», «Science Fiction.txt», «Ukrainian Literature.txt», «Psychology.txt», «Physics.txt», «Maths.txt», «Programming.txt», «Foreign Languages.txt», «Chemistry.txt», «Philosophy.txt», «Mechanics.txt» дані книжки такі як назва, автор, видавництво, рік, опис, сторінки, наявність, а для файлу «WorkerInfo.txt» зчитує дані про ПІБ працівника, посада та контактний номер і виводить у консоль.

Функція «SetInf», використовуючи файловий вивід <оfstream>, записує у файли

«Classic Fiction.txt», «Poetry.txt», «Fantasy.txt», «Detective.txt», «Novel.txt», «Science Fiction.txt», «Ukrainian Literature.txt», «Psychology.txt», «Physics.txt», «Maths.txt», «Programming.txt», «Foreign Languages.txt», «Chemistry.txt», «Philosophy.txt», «Mechanics.txt» дані книжки такі як назва, автор, видавництво, рік, опис, сторінки, наявність, а для файлу «WorkerInfo.txt» записує дані про ПІБ працівника, посада та контактний номер.

Функція «Menu», використовуючи потоковий вивід <ostream>, виводить у консоль інформацію про головне меню, а також про можливість завершити роботу.

Функція «FictionBookMenu», використовуючи потоковий вивід <ostream>, виводить у консоль інформацію про види книг, а саме: новели, українська література, класична проза, поезії, фантастику, наукову, детективи та перехід назад.

Функція «TechBookMenu», використовуючи потоковий вивід <ostream>, виводить у консоль інформацію про види книг, а саме: психологія, фізика, математика, програмування, іноземні мови, хімія, філософія, механіка та перехід назад.

## 3.2 Опис програмної реалізації алгоритму

### 3.2.1 Діаграма класів

Psychology

Physics

Maths

Programming

ForeignLanguage

Chemistry

Philosophy

Mechanics

ClassicProse

Poetry

Fantasy

ScienceFiction

Detective

Novel

UkrLiterature

**Book**

**Worker**

**Library**

**Рис.2 Діаграма класів**

Пояснення до діаграми:

Клас **Book** є батьківським класом, який містить такі обєкти:

* Ім’я (string Name)
* Автор (string Author)
* Видавництво (string Publisher)
* Рік (int year)
* Сторінки (int Pages)
* Наявність (bool Availability)
* Опис (string Description)

Дочірні класи **ClassicProse, Poetry, Fantasy, ScienceFiction, Detective, Novel, UkrLiterature, Psychology, Physics, Maths, Programming, ForeignLanguage, Chemistry, Philosophy і Mechanics** містять такі об’єкти як і батьківський клас:

* Ім’я (string Name)
* Автор (string Author)
* Видавництво (string Publisher)
* Рік (int year)
* Сторінки (int Pages)
* Наявність (bool Availability)
* Опис (string Description)

А також містять методи:

* Функція GetInf
* Функція SetInf

Клас **Library** містить такі об’єкти:

* Адреса (string Adress)
* Час роботи (string WorkingTimes)
* Номер телефону (long int Phone)

А також містить такі методи:

* Функція GetInf

Клас **Worker** містить такі об’єкти:

* Ім’я (string Name)
* Посада (string Job)
* Контактний номер (long Number)

А також містить такі методи:

* Функція GetInf

### 3.2.2. Схема викликів функції

Порядок викликів функцій:

1. Виведення головного меню (Menu)
2. Інформація про бібліотеку (Bibloteka.GetInf)
   1. Інформація про працівників (worker.GetInf)
   2. Додати працівника (worker.SetInf)
3. Інформація про книги (BookMenu)
   1. Художня література (FictionBookMenu)
   2. Технічна література (TechBookMenu)

# 4 Відлагодження та тестування реалізованої програми згідно індивідуального завдання

## 4.1 Опис структури програми

Я обрав **CppUnitTestFramework** для тестування своєї програми. Так як цей фреймворк вбудований у інтегральне середовище, де я розробляв свій проект, то мій погляд впав саме на ного.

## 4.2 Відлагодження та тестування окремих частин програми

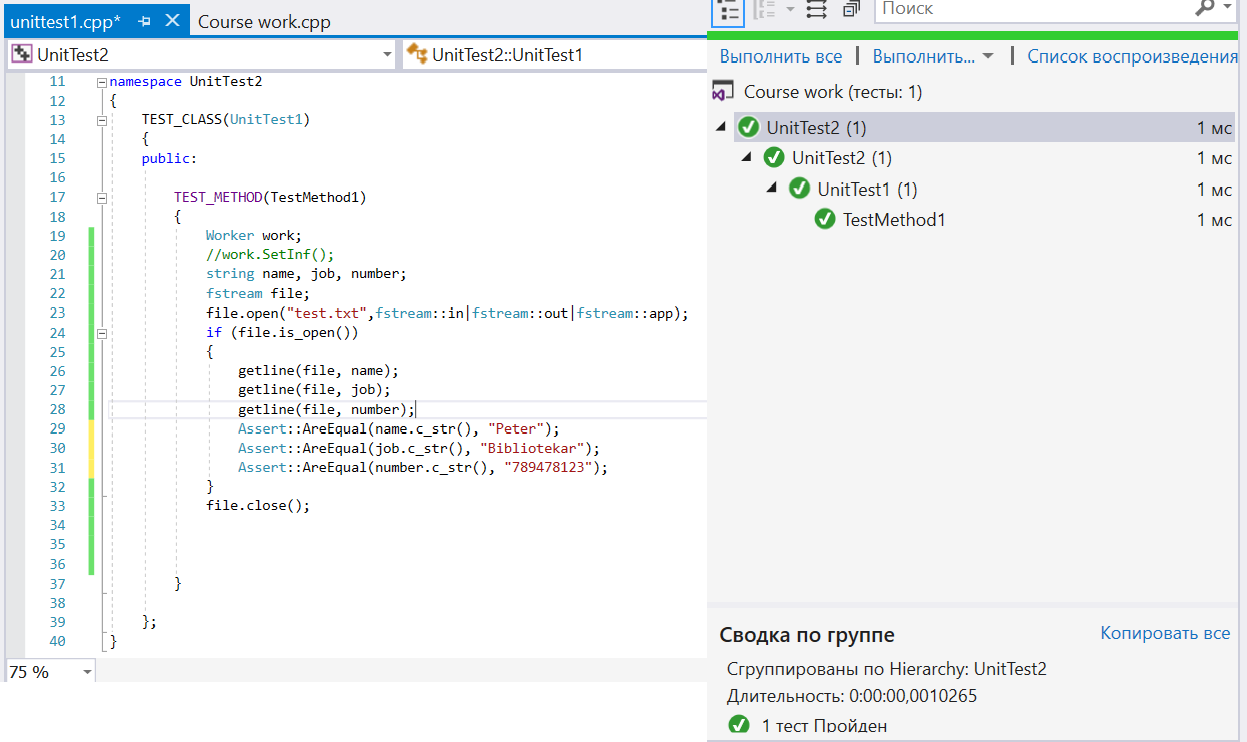
Для того, програма працювала потрібно використовувати середовище **Visual Studio 2017** (це середовище в якому я працював), версія операційної системи не принципова, але важливо мати достатньо ресурсів, тобто пам’ять та процесор, щоб програма працювала без перебоїв, я користуюсь **Windows 10 Home**. Щоб програма коректно працювала потрібно підключити бібліотеки без яких програма не буде запускатися, а саме: **«pch.h», «windows.h» «conio.h» «iostream» «fstream» «string»**, де без першої бібліотеки середовище не працює, друга – потрібна для коректного відображення української мови у консолі,третя - потрібна для коректного зчитування даних, четверта – потрібна для роботи з потоками виводу та вводу, п’ята – потрібна для роботи з файлами (відкрити, закрити, записати, змінити, видалити), шоста – потрібна для роботи із рядками. Також потрібно створити деякі файли, а саме папки: **ki, Books, Technical Literature, Fiction,** які знаходяться за таким адресом **C:\\ki\\Books\\ Fiction** або **C:\\ki\\Books\\ Technical Literature**, про все інше подбає сама програма, так як в основному це робота з файлами.

Юніт тести повинні перевіряти на правельність функції, які використовуються, тобто чи правильні значення вони повертають. Але у своїй роботі я використовую лише функції типу «**void»**, тобто функція не повертає ніяких значень, отже, я перевірятиму чи ця функція записує коректні значення у файл.

## 4.3 Відлагодження та тестування програми в цілому

Із розробкою своєї програми я стикався із багатьма помилками: логістичними, синтаксичними, але це була не проблема, бо з курсу програмування частина перша та друга ми ознайомлювались із більшістю помилок, які виникають при написані програми. Також у середовищі є вбудований відлагодувач, який показує помилки, які допустив користувач.

Результат виконання юніт тесту



**5.Відлагодження та тестування реалізованої програми згідно індивідуального завдання**

## 5.1. Аналіз структурної складності виконання програми

У контексті теоретичного аналізу центральним для інформатики є поняття складності. Неформально складність - це "інформаційна опірність середовища", тобто те, що заважає вирішувати завдання, принципово забороняє автоматизувати ті чи інші речі. Крім неформального визначення складності, теоретики інформатики придумали кілька набагато більше добре формалізованих видів складності, які фокусуються на більш вузьких властивостях завдань.

Перший такий вид складності - це ***ентропія,*** або кількість байт, потрібних для того, щоб описати задачу. Треба сказати, що назви збігаються не даремно, цей вид складності дивно схожий на термодинамічну ентропію як кількість можливих станів. Поняття інформаційної ентропії, яке використовує в своїй роботі К. Шеннон, в рамках конструктивістських ідей розглядається як "складність" структури, що лежить в основі системи (у тому числі і комунікаційної).

Другий вид складності - це "**заплутаність**" ***структур даних.***

Чисельність характеристики для нього немає, але для програміста він найпомітніше. Справа в тому, що більшість алгоритмів "побудовані" у вигляді маніпулювання якимись структурами даних. Часто ускладнення в програмі відбуваються за рахунок не самих алгоритмів, а за рахунок ускладнення структур, що представляють дані . Крім того, ієрархічним ускладнення схильні не тільки структури даних, а й алгоритми роботи з ними. Віртуальна машина, використовувана програмістом при написанні програм на деякій мові високого рівня, є ієрархічною системою віртуальних машин . Важливим з точки зору розвитку комп'ютерних технологій стає цей перехід до мов "високого рівня". З цього моменту програмування дійсно стає діяльністю зі створення складних систем.

Третій вид складності - **колмогорівска**.

Вона являє собою математично оформлений аналог "заплутаність". Наявність або відсутність великих обсягів роботи і складності моделі не вичерпують принципових проблем, з якими стикаються люди при створенні автономних систем. Є ще якась досить неочевидна характеристика, яка говорить нам, наскільки взагалі задача, яку ми намагаємося вирішити, піддасться алгоритмізації. Ця характеристика має числову величину і називається ***колмогорівскою складністю.***

Дуже цікаво, що колмогоровской складність має зв'язок з ентропією Шеннона, згаданої на початку глави. І та і інша в деякому розумінні допомагають формалізувати задачу "зниження рівня невизначеності".

Якщо з цих позицій підійти до поняття обчислення з точки зору математики, то "складність" в сенсі алгорітмізіруемості - це складність щодо оптимального мови, вимірюють довжину найкоротшого опису цього обчислення. Чим складніше об'єкт, тим більше інформації він містить. Ця "довжина опису", що задає модель обчислень, яка описується як довжина довічних слів (технічно - кінцеві послідовності бітів) і називається " ***колмогорівскою складністю*** ".

Тобто на інженерному рівні складність і буде описом способу конструювання, хоча на рівні теорії вичіслімих функцій воназадається функціонально: "... спосіб опису є обчислювана функція з множин довічних слів".

У певному сенсі, в математиці колмогоровской складності доведено, що "для випадкових даних" колмогоровской складність (довжина програми, що моделює ці дані) буде приблизно дорівнює ентропії Шеннона цих даних. Проте показано, що для невипадкових даних колмогоровской складність мала, що, власне, і дозволяє програмістам писати програми, що моделюють різні системи .

Четвертий різновид складності - ***обчислювальна.***

Складність обчислювальних процесів — це поняття теорії складності обчислень, оцінка ресурсів (зазвичай часу) необхідних для виконання алгоритму. Для оцінки алгоритмів існує багато критеріїв. Найбільшу увагу приділяють порядку росту необхідних для розв'язання задачі часу та розміру пам'яті при збільшенні розміру вхідних даних.

П’ятий різновид складності – ***часова.***

Часова складність алгоритму в комп'ютерних науках є обчислювальною складністю алгоритму, яка описує час потрібний для виконання алгоритму. Вона зазвичай визначається шляхом підрахунку кількості елементарних операцій, виконуваних алгоритмом, при цьому вважають, що кожна елементарна операція виконується за фіксовану кількість часу. Таким чином, кількість часу і кількість елементарних операцій, необхідних для виконання алгоритму, відрізняються постійним множником.

## 5.2. Аналіз часу виконання програми в залежності від об'єму вхідних даних

1). При запуску програми відкривається меню (Рис.3):

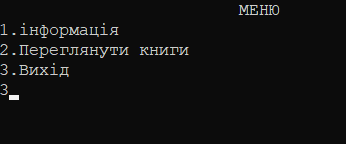


Рис. 3

2). Обравши 3, програма закриється (Рис. 4):

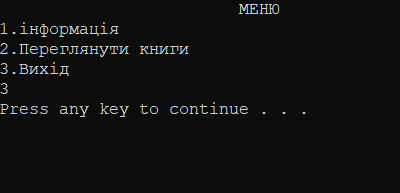
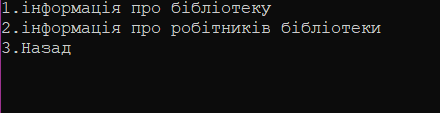


Рис. 4

3).Обравши 1, відкриється меню з подальшим вибором (Рис.5):



4).Обравши 1, відкриється інформація про бібліотеку (Рис. 6):

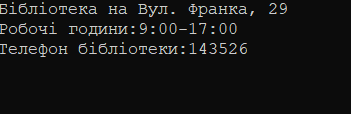


Рис.6

5). Повернувшись назад до меню з інформацією та обравши п.2 , відкриється меню, в якому можна буде переглянути працівників, або додати їх(Рис.7):

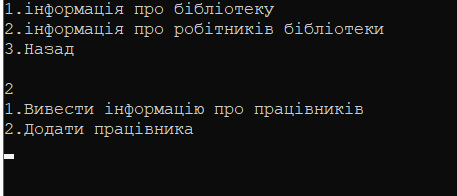


Рис.7

6).Обравши п.2, в нас з’являється можливість заповнити необхідні дані і внести їх в файл, в якому зберігається інформація про всіх робітників бібліотеки(Рис. 8):

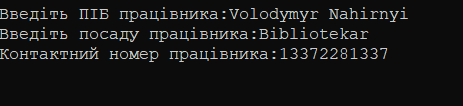


Рис. 8

7). Повернувшись назад до меню з інформацією про працівників та обравши п.1, відкриється меню з інформацією, заповненої на рис.8, це можна побачити на Рис.9

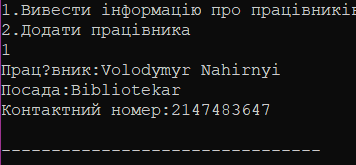
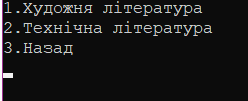
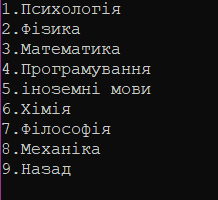


Рис.9

8). Повернувшись до головного меню та обравши п.2, відкриється меню, в якому треба обрати групу книг, візьмемо наприклад Технічну літературу.(рис.10)



9). Після того, як обрали групу книг, відкривається меню, в якому треба вибрати відповідний жанр книги, візьмемо, наприклад, Програмування(п.4) (рис.11):



10).Відкриється меню, в якому можна або передивитися вже внесені в файл книги, або додати нові. Спробуємо вивести всі книги даного жанру на екран(рис.12):

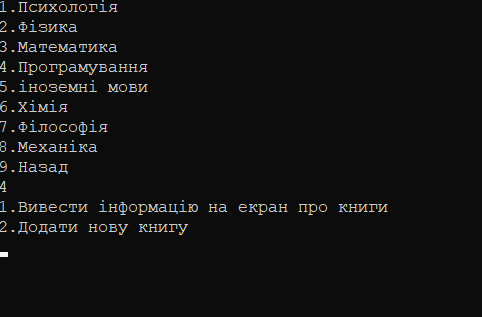


Рис.12

11).Так як такий файл був видалений попередньо, програма це побачила і тепер просить користувача, чи хоче він створити такий файл(рис. 13)

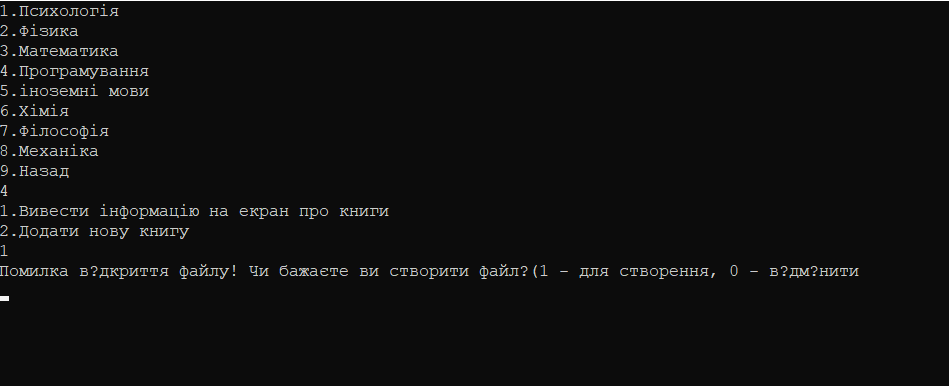
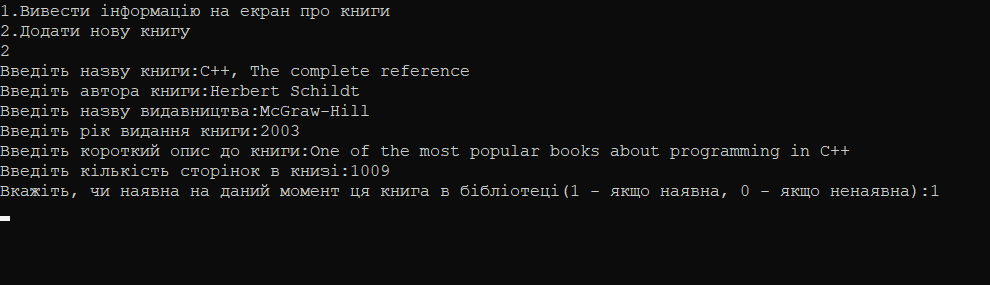


Рис.13

12).Обравши п.2, програма створить відповідний файл, який в подальшому можна буде переглядати в пошуку книг. Повернувшись назад до меню з вибором, оберемо п.2 і додамо нову книгу в даний файл (рис.14):



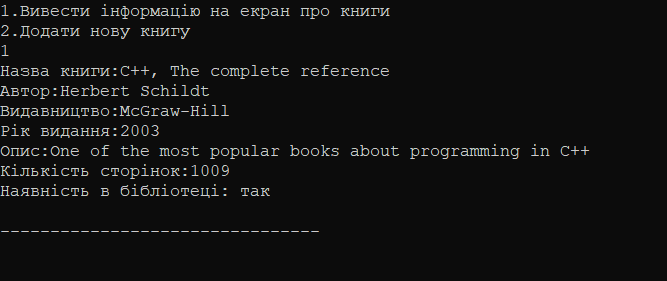
13).Програма занесе дану інформацію в файл, і її можна буде зчитати, обравши п.2 в відповідному меню(рис.15):

Рис.15

# Висновок

У процесі виконання даної курсової роботи на прикладі реалізації такої предметної галузі як бібліотека, я закріпив навички об’єктно-орієнтованого програмування, яких набув під час вивчення даного предмету у 2 семестрі.

Також закріпив на практиці використання об’єктно-орієнтованого підходу, використання основних принципів ООП, використання можливостей мови програмування С++.

З переваг моєї програми можна виділити простоту використання, інтуїтивність, висока швидкодія через використання консольного інтерфейсу, перевірка правильності вводу, наявності всіх файлів з інформацією та коректності використання програми.

З недоліків моєї програми можна виділити використання консольного інтерфейсу, що є застарілим для використання програм даного типу та те, що вона не охоплює всі аспекти такої предметної галузі, як бібліотека, себто, має не надто високий функціонал в порівнянні з справжньою бібліотекою.

# Використана література:

1. Р.Лафоре .Об’єктно-орієнтовне програмування в С++. / Роберт Лафоре // Programmer’s Klondike. – 2010. - № 4. Режим доступу:

<https://proklondike.net/books/cpp/lafore_oop_v_cpp.html>

1. М.Вайсфелд. Объектно-ориентированное мышление. / Мэтт Вайсфельд // Флибуста. – 2016. № 1. Режим доступу:

<http://flibusta.site/b/450076>

1. Г.Буч. Объектно-ориентированный анализ и проектирование с примерами приложений. / Граді Буч // Programmer’s Klondike. - 2006. № 1. Режим доступу:

<https://proklondike.net/books/oop/buch_ooad.html>

**Додаток**

#include "pch.h"

#include <iostream>

#include <fstream>

#include <string>

#include <conio.h>

#include"windows.h"

using namespace std;

int choice;

class Book {

protected:

string Name;

string Author;

string Publisher;

int year;

int Pages;

bool Availability;

string Description;

public:

};

class ClassicProse : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Fiction\\Classic Fiction.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Classic Fiction.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Fiction\\Classic Fiction.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Classic Fiction.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Poetry : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Fiction\\Poetry.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Poetry.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Fiction\\Poetry.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Poetry.txt");

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

}

};

class Fantasy : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Fiction\\Fantasy.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Fantasy.txt");

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Fiction\\Fantasy.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Fantasy.txt");

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

}

};

class ScienceFiction : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Fiction\\Science Fiction.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Science Fiction.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Fiction\\Science Fiction.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Science Fiction.txt");

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

}

};

class Detective : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Fiction\\Detective.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Detective.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Fiction\\Detective.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Detective.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Novel : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Fiction\\Novel.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Novel.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Fiction\\Novel.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Novel.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class UkrLiterature : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Fiction\\Ukrainian Literature.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Ukrainian Literature.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Fiction\\Ukrainian Literature.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Fiction\\Ukrainian Literature.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Psychology : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Psychology.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Psychology.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Psychology.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Psychology.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Physics : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Physics.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Physics.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Physics.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Physics.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Maths : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Maths.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Maths.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Maths.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Maths.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Programming : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Programming.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Programming.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Programming.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Programming.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class ForeignLanguage : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Foreign Languages.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Foreign Languages.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Foreign Languages.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Foreign Languages.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Chemistry : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Chemistry.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Chemistry.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Chemistry.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Chemistry.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Philosophy : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Philosophy.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Philosophy.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Philosophy.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Philosophy.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Mechanics : public Book {

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Mechanics.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Mechanics.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\Technical Literature\\Mechanics.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\Technical Literature\\Mechanics.txt");

}

}

else {

cout << "Введiть назву книги:";

cin.get();

getline(cin, Name);

file << "Назва книги:" << Name << endl;

cout << "Введiть автора книги:";

getline(cin, Author);

file << "Автор:" << Author << endl;

cout << "Введiть назву видавництва:";

getline(cin, Publisher);

file << "Видавництво:" << Publisher << endl;

cout << "Введiть рiк видання книги:";

cin >> year;

file << "Рiк видання:" << year << endl;

cout << "Введiть короткий опис до книги:";

cin.get();

getline(cin, Description);

file << "Опис:" << Description << endl;

cout << "Введiть кiлькiсть сторiнок в книзi:";

cin >> Pages;

file << "Кiлькiсть сторiнок:" << Pages << endl;

cout << "Вкажiть, чи наявна на даний момент ця книга в бiблiотецi(1 - якщо наявна, 0 - якщо ненаявна):";

cin >> Availability;

if (!Availability) {

file << "Наявнiсть в бiблiотецi: нi" << endl;

}

else {

file << "Наявнiсть в бiблiотецi: так" << endl;

}

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

class Library {

private:

string Adress;

string WorkingTimes;

long int Phone;

public:

Library(string Adress, string WorkingTimes, long int Phone) {

this->Adress = Adress;

this->WorkingTimes = WorkingTimes;

this->Phone = Phone;

}

void GetInf() {

system("cls");

cout << "Бiблiотека на " << Adress << endl;

cout << "Робочi години:" << WorkingTimes << endl;

cout << "Телефон бiблiотеки:" << Phone << endl;

}

};

class Worker {

private:

string Name;

string Job;

long Number;

public:

void GetInf() {

string Info;

ifstream file;

file.open("C:\\ki\\Books\\WorkerInfo.txt");

if (!file.is\_open()) {

cout << "Помилка відкриття файлу! Чи бажаєте ви створити файл?(1 - для створення, 0 - відмінити" << endl;

cin >> choice;

if (choice == 1) {

ofstream outfile("C:\\ki\\Books\\WorkerInfo.txt");

}

}

else {

while (getline(file, Info)) {

cout << Info << endl;

}

}

}

void SetInf() {

ofstream file;

file.open("C:\\ki\\Books\\WorkerInfo.txt", ios\_base::app);

if (!file.is\_open()) {

cout << "Error!" << endl;

}

else {

cout << "Введiть ПIБ працiвника:";

cin.get();

getline(cin, Name);

file << "Працівник:" << Name << endl;

cout << "Введiть посаду працiвника:";

getline(cin, Job);

file << "Посада:" << Job << endl;

cout << "Контактний номер працiвника:";

cin >> Number;

file << "Контактний номер:" << Number << endl;

file << endl << "--------------------------------" << endl;

file.close();

}

}

};

int StateMenu;

void Menu() {

system("cls");

cout << "\t\t\tМЕНЮ" << endl;

cout << "1.iнформацiя" << endl;

cout << "2.Переглянути книги" << endl;

cout << "3.Вихiд" << endl;

cin >> StateMenu;

}

void FictionBookMenu() {

system("cls");

cout << "1.Класична проза" << endl;

cout << "2.Поезiя" << endl;

cout << "3.Фентезi" << endl;

cout << "4.Фантастика" << endl;

cout << "5.Детектив" << endl;

cout << "6.Роман" << endl;

cout << "7.Українська лiтература" << endl;

cout << "8. Назад"<<endl;

}

void TechBookMenu() {

system("cls");

cout << "1.Психологiя" << endl;

cout << "2.Фiзика" << endl;

cout << "3.Математика" << endl;

cout << "4.Програмування" << endl;

cout << "5.iноземнi мови" << endl;

cout << "6.Хiмiя" << endl;

cout << "7.Фiлософiя" << endl;

cout << "8.Механiка" << endl;

cout << "9.Назад" << endl;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

Worker worker;

bool a = true;

Library Bibloteka("Вул. Франка, 29", "9:00-17:00", 143526);

ClassicProse BookClassicProse;

Poetry BookPoetry;

Fantasy BookFantasy;

ScienceFiction BookScienceFiction;

Detective BookDetective;

Novel BookNovel;

UkrLiterature BookUkrLiterature;

Psychology BookPsychology;

Physics BookPhysics;

Maths BookMaths;

Programming BookProgramming;

ForeignLanguage BookForeignLanguage;

Chemistry BookChemistry;

Philosophy BookPhilosophy;

Mechanics BookMechanics;

Menu();

while (a) {

if (StateMenu == 1) {

system("cls");

cout << "1.iнформацiя про бiблiотеку" << endl;

cout << "2.iнформацiя про робiтникiв бiблiотеки" << endl;

cout << "3.Назад" << endl;

cin >> choice;

switch (choice) {

case 1:

Bibloteka.GetInf();

\_getch();

break;

Menu();

case 2:

cout << "1.Вивести iнформацiю про працiвникiв" << endl;

cout << "2.Додати працiвника" << endl;

cin >> choice;

switch (choice) {

case 1:

worker.GetInf();

\_getch();

break;

Menu();

case 2:

worker.SetInf();

\_getch();

break;

Menu();

}

break;

case 3:

Menu();

}

}

else if

(StateMenu == 2) {

system("cls");

cout << "1.Художня лiтература" << endl;

cout << "2.Технiчна лiтература" << endl;

cout << "3.Назад" << endl;

cin >> choice;

switch (choice) {

case 1:

FictionBookMenu();

cin >> choice;

switch (choice) {

case 1:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookClassicProse.GetInf();

\_getch();

Menu();

break;

case 2:

BookClassicProse.SetInf();

\_getch();

break;

Menu();

}

break;

case 2:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookPoetry.GetInf();

\_getch();

break;

Menu();

case 2:

BookPoetry.SetInf();

\_getch();

break;

Menu();

}

break;

case 3:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookFantasy.GetInf();

\_getch();

break;

Menu();

case 2:

BookFantasy.SetInf();

\_getch();

break;

Menu();

}

break;

case 4:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookScienceFiction.GetInf();

\_getch();

break;

Menu();

case 2:

BookScienceFiction.SetInf();

\_getch();

break;

Menu();

}

break;

case 5:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookDetective.GetInf();

\_getch();

break;

Menu();

case 2:

BookDetective.SetInf();

\_getch();

break;

Menu();

}

break;

case 6:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookNovel.GetInf();

\_getch();

break;

Menu();

case 2:

BookNovel.SetInf();

\_getch();

break;

Menu();

}

break;

case 7:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookUkrLiterature.GetInf();

\_getch();

break;

Menu();

case 2:

BookUkrLiterature.SetInf();

\_getch();

break;

Menu();

}

break;

case 8:

\_getch();

Menu();

}

break;

case 2:

TechBookMenu();

cin >> choice;

switch (choice) {

case 1:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookPsychology.GetInf();

\_getch();

break;

Menu();

case 2:

BookPsychology.SetInf();

\_getch();

break;

Menu();

}

break;

case 2:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookPhysics.GetInf();

\_getch();

break;

Menu();

case 2:

BookPhysics.SetInf();

\_getch();

break;

Menu();

}

break;

case 3:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookMaths.GetInf();

\_getch();

break;

Menu();

case 2:

BookMaths.SetInf();

\_getch();

break;

Menu();

}

break;

case 4:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookProgramming.GetInf();

\_getch();

break;

Menu();

case 2:

BookProgramming.SetInf();

\_getch();

break;

Menu();

}

break;

case 5:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookForeignLanguage.GetInf();

\_getch();

break;

Menu();

case 2:

BookForeignLanguage.SetInf();

\_getch();

break;

Menu();

}

break;

case 6:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookChemistry.GetInf();

\_getch();

break;

Menu();

case 2:

BookChemistry.SetInf();

\_getch();

break;

Menu();

}

break;

case 7:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookPhilosophy.GetInf();

\_getch();

break;

Menu();

case 2:

BookPhilosophy.SetInf();

\_getch();

break;

Menu();

}

break;

case 8:

cout << "1.Вивести iнформацiю на екран про книги" << endl;

cout << "2.Додати нову книгу" << endl;

cin >> choice;

switch (choice) {

case 1:

BookMechanics.GetInf();

\_getch();

break;

Menu();

case 2:

BookMechanics.SetInf();

\_getch();

break;

Menu();

}

break;

case 9:

break;

Menu();

}

case 3:

Menu();

}

}

else if (StateMenu == 3) {

a = false;

}

}

return 0;

}