INTRODUCTION

NAME:

- RAO ALI AHMAD

  223710

- M UMAIR LATIF

  223644

oop

PROJECT REPORT

SUBMITTED TO:

Dr. shaukat hayat

Air university multan campus

CONTENTS:

TOPIC:

PROJECT REPORT

Software (os,language&packages)used
1) operating system : DOS
2) language: c++
3) packages: dev c++

ACKNOWLEGEMENT

There is much difference between saying and doing. So, to get theoretical knowledge is not enough. It provides only base. we So, we make a project namely " book store management" we put my lots of effort to
do this project and try my best.

We believe that "Practical leads a man towards Performance"

In this project we thanks to our colleges which helps to motivate us.

At last we thankful to the DR SHAUKAT HAYAT who help us to decide this project and encourage us.

Report: Bookstore Management System

Introduction: The Bookstore Inventory Management System is designed to manage the inventory of books in a bookstore. It utilizes object-oriented programming concepts such as inheritance, polymorphism, and error handling to provide a flexible and efficient solution for book inventory management.

Learning Outcomes: Through this project, the following learning outcomes can be achieved:

1.Understanding the concept of inheritance and its practical implementation in OOP.

2.Familiarity with polymorphism and its use to treat objects of different classes as objects of a common base class.

3.Knowledge of virtual functions and their role in enabling late binding and dynamic dispatch.

4. Comprehension of abstract classes and their significance in providing a structure for derived classes and enforcing a certain contract.

Objectives: The main objectives of the Bookstore Management System are as follows:

- To demonstrate the use of inheritance to create specialized book types derived from a base class.
- To showcase polymorphism by treating different book objects as instances of the common base class and calling their specific functions.
- To implement virtual functions to enable late binding and dynamic dispatch of function calls.
- To create an abstract class that serves as a blueprint for derived classes and enforces specific behavior.

Class Design: 2.1. Base Class - Book: The base class "Book" is an abstract class that serves as the blueprint for all types of books in the inventory. It has the following attributes:

title: Stores the title of the book.
author: Stores the author of the book.
price: Stores the price of the book.
quantity: Stores the quantity of the book in stock.
The "display()" function is declared as a pure virtual function in the base class, which means it must be implemented in the derived classes. This allows for polymorphic behavior when displaying book information.

2.2. Derived Classes - Fiction and Non Fiction: The derived classes "Fiction" and "NonFiction" inherit from the base class "Book." These classes can add additional attributes or behaviors specific to each book type. For example, the "Fiction" class could have attributes like genre or protagonist, while the "NonFiction" class could have attributes like subject or ISBN.
Both the "Fiction" and "NonFiction" classes implement the "display()" function to display the relevant information for each book type. The implementation of the

"display()" function in each class can utilize the base class attributes and any additional attributes specific to the derived class.

2.3. Bookstore Class: The "Bookstore" class is responsible for managing the inventory of books. It uses a data structure, such as a list or array, to store instances of the "Book" objects. The class provides the following functionalities:
Adding books to the inventory: This function takes user inputs for the book details (title, author, price, quantity) and creates an instance of the appropriate derived class (Fiction or Non Fiction). It then adds the book to the inventory.
Displaying the inventory: This function calls the "display()" function for each book in the inventory, which invokes the appropriate version of the function based on the actual object type (polymorphism).

Polymorphism and Inheritance: To demonstrate polymorphism, objects of the derived classes (Fiction and Non Fiction) can be created and stored as base class pointers. For example:

```
Book* book1 = new Fiction("The Great Gatsby", "F. Scott Fitzgerald", 10.
Book* book2 = new NonFiction("Sapiens: A Brief History of Humankind", "Y
```

These objects can then be added to the bookstore's inventory using the appropriate functions, and the display function can be called to observe the polymorphic behavior.

User input validation: When adding a book to the inventory, the system validates user inputs for book price and quantity to ensure they are within acceptable ranges. If the inputs are invalid, appropriate error messages are displayed, and the user is prompted to enter valid value.

Conclusion: The Bookstore Inventory Management System provides an efficient and flexible solution for managing the inventory of books in a bookstore. By utilizing object-oriented concepts such as inheritance, polymorphism, and error handling, the system allows for easy addition of new book types and provides a user-friendly interface for managing the bookstore's inventory effectively.