INSTITUTE OF TECHNOLOGY AND MANAGEMENT
SKILLS UNIVERSITY,
KHARGHAR, NAVI MUMBAI

PYTHON PROGRAMMING LAB

Prepared by:

Name of Student: Shikha singh________________________

Roll No: ___25_________

Batch: 2023-27

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

xp. No	List of Experiment
	1.1 Write a program to compute Simple Interest.
	1.2 Write a program to perform arithmetic, Relational operators.
	1.3 Write a program to find whether a given no is even & odd.
	1.4 Write a program to print first n natural number & their sum.
	1.5 Write a program to determine whether the character entered is a Vowel or not .
	1.6 Write a program to find whether given number is an Armstrong Number.
	1.7 Write a program using for loop to calculate factorial of a No.
	1.8 Write a program to print the following pattern
	i) * * * * * * * * * * * * * * *
	ii) 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
	iii) *

	2.1 Write a program that define the list of defines the list of define countries that are in BRICS.
	2.2 Write a program to traverse a list in reverse order. 1.By using Reverse method. 2.By using slicing
	2.3 Write a program that scans the email address and forms a tuple of username and domain.
	2.4 Write a program to create a list of tuples from given list having number and add its cube in tuple. i/p: c= [2,3,4,5,6,7,8,9]
	2.5 Write a program to compare two dictionaries in Python? (By using == operator)
	2.6 Write a program that creates dictionary of cube of odd numbers in the range.
	2.7 Write a program for various list slicing operation. a= [10,20,30,40,50,60,70,80,90,100] i. Print Complete list ii. Print 4th element of list iii. Print list from0th to 4th index. iv. Print list -7th to 3rd element v. Appending an element to list. vi. Sorting the element of list. vii. Popping an element. viii. Removing Specified element. ix. Entering an element at specified index. x. Counting the occurrence of a specified element. xi. Extending list. xii. Reversing the list.
	3.1 Write a program to extend a list in python by using given approach. i. By using + operator. ii. By using Append ()

	iii. By using extend ()
	3.2 Write a program to add two matrices.
	3.3 Write a Python function that takes a list and returns a new list with distinct elements from the first list.
	3.4 Write a program to Check whether a number is perfect or not.
	3.5 Write a Python function that accepts a string and counts the number of upper- and lower-case letters. string_test= 'Today is My Best Day'
	4.1 Write a program to Create Employee Class & add methods to get employee details & print.
	4.2 Write a program to take input as name, email & age from user using combination of keywords argument and positional arguments (*args and**kwargs) using function,
	4.3 Write a program to admit the students in the different Departments(pgdm/btech) and count the students. (Class, Object and Constructor).
	4.4 Write a program that has a class store which keeps the record of code and price of product display the menu of all product and prompt to enter the quantity of each item required and finally generate the bill and display the total amount.
	4.5 Write a program to take input from user for addition of two numbers using (single inheritance).
	4.6 Write a program to create two base classes LU and ITM and one derived class. (Multiple inheritance).
	4.7 Write a program to implement Multilevel inheritance, Grandfather□Father-□Child to show property inheritance from grandfather to child.
	4.8 Write a program Design the Library catalogue system using inheritance take base class (library item) and derived class (Book, DVD & Journal) Each derived

	class should have unique attribute and methods and system should support Check in and check out the system. (Using Inheritance and Method overriding)
	5.1 Write a program to create my_module for addition of two numbers and import it in main script.
	5.2 Write a program to create the Bank Module to perform the operations such as Check the Balance, withdraw and deposit the money in bank account and import the module in main file.
	5.3 Write a program to create a package with name cars and add different modules (such as BMW, AUDI, NISSAN) having classes and functionality and import them in main file cars.
	6.1 Write a program to implement Multithreading. Printing “Hello” with one thread & printing “Hi” with another thread.
.	7.1 Write a program to use ‘whether API’ and print temperature of any city, also print the sunrise and sunset times for the same humidity of that area.
	7.2 Write a program to use the ‘API’ of crypto currency.

Name of Student: _Shikha singh________________

Roll Number: _25________________

Experiment No: 4.8

Title:

4.8 Write a program Design the Library catalogue system using inheritance take

base class (library item) and derived class (Book, DVD &amp; Journal) Each derived

class should have unique attribute and methods and system should support Check

in and check out the system. (Using Inheritance and Method overriding)class should have unique attribute and methods and system should support Check in and check out the system. (Using Inheritance and Method overriding)

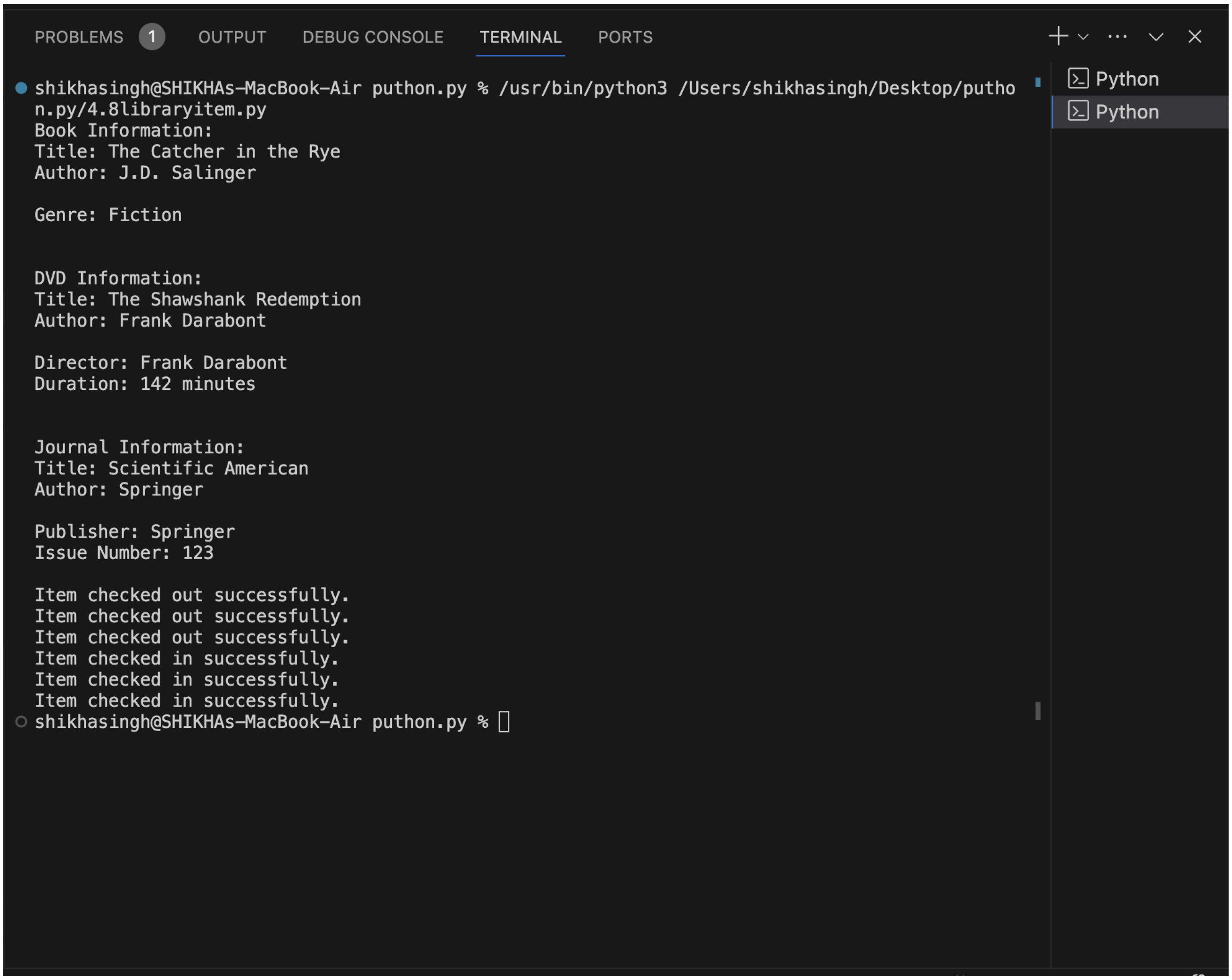
Theory:

- Base Class (LibraryItem): The LibraryItem class serves as the base class, containing common attributes and methods shared by all library items, such as title, author, and methods for displaying information, checking in, and checking out.
- Derived Classes (Book, DVD, Journal): These classes inherit from the base class LibraryItem and extend its functionality by adding attributes specific to each type of library item (e.g., genre for Book, director and duration for DVD, publisher and issue number for Journal). Each derived class also overrides the displayInfo method to provide specialized information.
- Method Overriding: The concept of method overriding is demonstrated in the derived classes. The displayInfo method in each derived class overrides the method in the base class, allowing for custom behavior for displaying information relevant to that particular type of library item.
- Check In and Check Out System: The base class provides generic methods for checking in and checking out items. Each derived class can override these methods to implement specific behavior. For example, a book may have a different check-out process than a DVD.

Code:

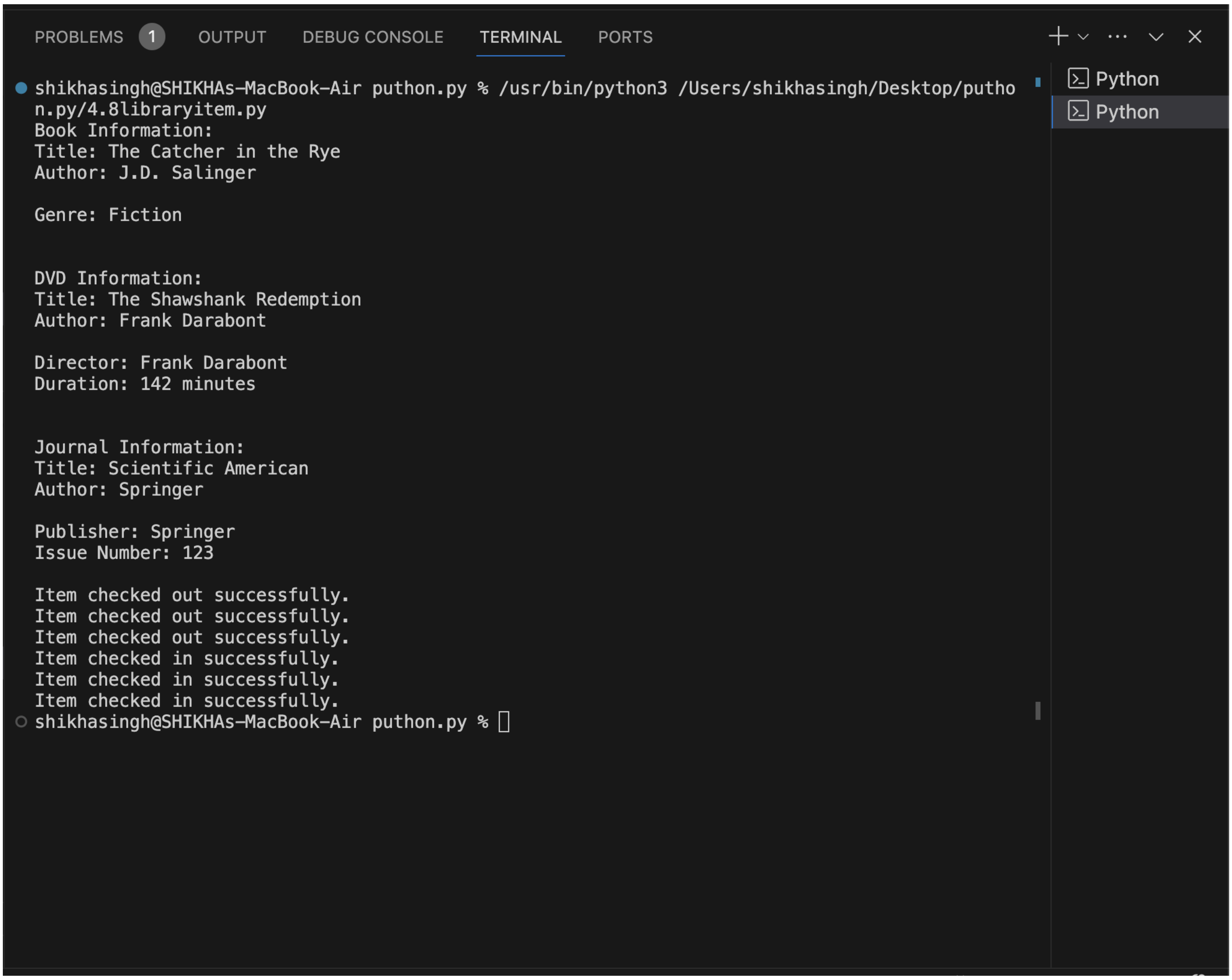
```
class LibraryItem:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.checked_out = False
    def display_info(self):
        print(f"Title: {self.title}\nAuthor: {self.author}\n")
    def check_out(self):
        self.checked_out = True
        print("Item checked out successfully.")
    def check_in(self):
        self.checked_out = False
        print("Item checked in successfully.")
class Book(LibraryItem):
    def __init__(self, title, author, genre):
        super().__init__(title, author)
        self.genre = genre
    def display_info(self):
        super().display_info()
        print(f"Genre: {self.genre}\n")
class DVD(LibraryItem):
    def __init__(self, title, director, duration):
        super().__init__(title, director)
        self.director = director
        self.duration = duration
    def display_info(self):
        super().display_info()
        print(f"Director: {self.director}\nDuration: {self.duration} minutes\n")
class Journal(LibraryItem):
    def __init__(self, title, publisher, issue_number):
        super().__init__(title, publisher)
        self.publisher = publisher
        self.issue_number = issue_number
    def display_info(self):
        super().display_info()
        print(f"Publisher: {self.publisher}\nIssue Number: {self.issue_number}\n")
book = Book("The Catcher in the Rye", "J.D. Salinger", "Fiction")
dvd = DVD("The Shawshank Redemption", "Frank Darabont", 142)
journal = Journal("Scientific American", "Springer", 123)
print("Book Information:")
book.display_info()
print("\nDVD Information:")
dvd.display_info()
print("\nJournal Information:")
journal.display_info()
book.check_out()
dvd.check_out()
journal.check_out()
book.check_in()
dvd.check_in()
journal.check_in()
```

Output: (screenshot)



```
shikhasingh@SHIKHAs-MacBook-Air puthon.py % /usr/bin/python3 /Users/shikhasingh/Desktop/putho
n.py/4.8libraryitem.py
Book Information:
Title: The Catcher in the Rye
Author: J.D. Salinger

Genre: Fiction


DVD Information:
Title: The Shawshank Redemption
Author: Frank Darabont

Director: Frank Darabont
Duration: 142 minutes


Journal Information:
Title: Scientific American
Author: Springer

Publisher: Springer
Issue Number: 123

Item checked out successfully.
Item checked out successfully.
Item checked out successfully.
Item checked in successfully.
Item checked in successfully.
Item checked in successfully.
shikhasingh@SHIKHAs-MacBook-Air puthon.py % 
```

Test Case: Any two (screenshot)



```
shikhasingh@SHIKHAs-MacBook-Air puthon.py % /usr/bin/python3 /Users/shikhasingh/Desktop/putho
n.py/4.8libraryitem.py
Book Information:
Title: The Catcher in the Rye
Author: J.D. Salinger

Genre: Fiction


DVD Information:
Title: The Shawshank Redemption
Author: Frank Darabont

Director: Frank Darabont
Duration: 142 minutes


Journal Information:
Title: Scientific American
Author: Springer

Publisher: Springer
Issue Number: 123

Item checked out successfully.
Item checked out successfully.
Item checked out successfully.
Item checked in successfully.
Item checked in successfully.
Item checked in successfully.
shikhasingh@SHIKHAs-MacBook-Air puthon.py % 
```

Conclusion:

The use of inheritance in the library catalog system design provides a structured and extensible solution. The base class encapsulates common functionality, promoting code reuse and reducing redundancy. Derived classes inherit and extend this functionality, allowing for specialization in handling different types of library items.

The approach taken in the code demonstrates the flexibility of OOP, making it easier to add new types of library items in the future without modifying existing code significantly. The use of method overriding in derived classes ensures that the appropriate version of a method is called based on the type of object, allowing for polymorphic behavior.

In summary, the implementation adheres to OOP principles, making the code modular, maintainable, and scalable for potential future enhancements in the library catalog system.