Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 41
FP Practical 9

Institute of Computer Technology
B. Tech Computer Science and Engineering

Sub: (2CSE403) FUNCTIONAL PROGRAMMING

Practical 9

1. Design a library application based on the following requirements. Each book in the library has related title and author. All the books in library are soft-books which imply that book may exist as either an EBook or Audio book. Every EBook has: Format: PDF, EPUB, MOBI, AZW. Anything else supplied as format should give compilation error. Every EBook has number of pages. Audiobook has: Track length measured in minutes. Format: MP3, WMA, WAV. Anything else supplied as format should give compilation error. Application should be able to display books' name, author, format and description if required

Code :

```
from YSL_io import *


class Book:


   def __init__(self, title: str, author: str):
       self.title = title
       self.author = author


class EBook(Book):
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 41
FP Practical 9

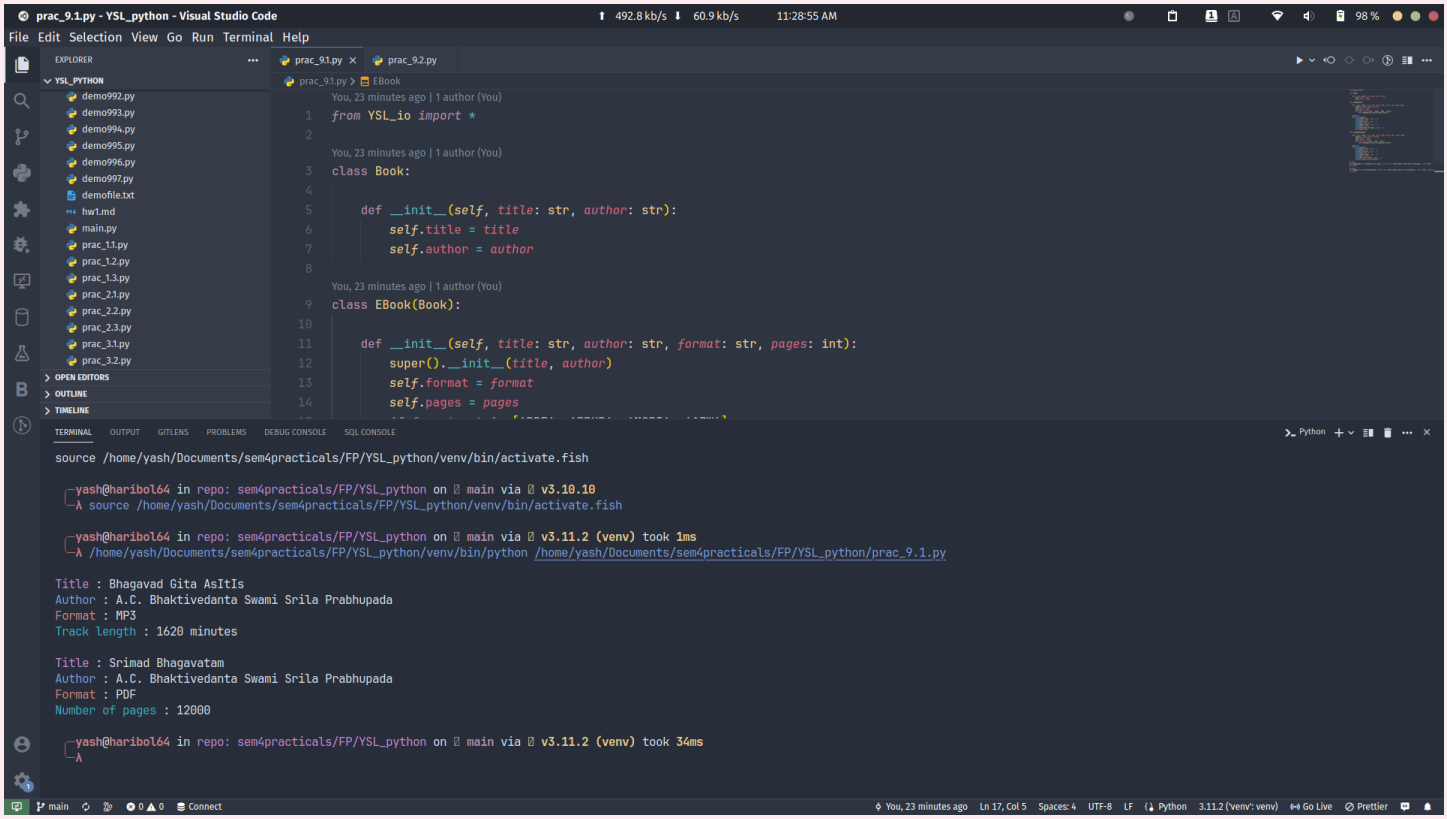
```
    def __init__(self, title: str, author: str, format: str, pages: int):
        super().__init__(title, author)
        self.format = format
        self.pages = pages
        if format not in ['PDF', 'EPUB', 'MOBI', 'AZW']:
            raise ValueError("Invalid EBook format")

    @property
    def display(self):
        printMGNTA('Title', end=' : ')
        print(self.title)
        printBLU('Author', end=' : ')
        print(self.author)
        printORNG('Format', end=' : ')
        print(self.format)
        printGRN('Number of pages', end=' : ')
        print(self.pages)

class Audiobook(Book):

    def __init__(self, title: str, author: str, format: str, length: int):
        super().__init__(title, author)
        self.format = format
        self.length = length
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 41
FP Practical 9

```
        if format not in ['MP3', 'WMA', 'WAV']:
            raise ValueError("Invalid Audiobook format")

    @property
    def display(self):
        printMGNTA('Title', end=' : ')
        print(self.title)
        printBLU('Author', end=' : ')
        print(self.author)
        printORNG('Format', end=' : ')
        print(self.format)
        printGRN('Track length', end=' : ')
        print(f'{self.length} minutes')

print()
bg = Audiobook(title='Bhagavad Gita AsItIs', author='A.C. Bhaktivedanta
Swami Srila Prabhupada', format='MP3', length=1620)
bg.display

print()
sb = EBook(title='Srimad Bhagavatam', author='A.C. Bhaktivedanta Swami
Srila Prabhupada', format='PDF', pages=12000)
sb.display
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 41
FP Practical 9

Output :



2. Implement Stack using concept of Object oriented programming. At minimum, any stack should be able to perform the following three operations: Push: Add an object passed as an argument to the top of the stack. Pop: Remove the object at the top of the stack and return it. Peek (or peep): Return the object at the top of the stack (without removing it). Display: Print the current status of stack.

Code :

```
from YSL_io import *

class Stack:

    def __init__(self, l1=[]):
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 41
FP Practical 9

```
        self.stck = l1

    def Push(self, y):
        try:
            self.stck.append(y)
        except:
            printRED('Something went wrong!')
        else:
            print(y, end=' ')
            printGRN('pushed to the stack')
            self.Display()

    def Pop(self):
        try:
            y = self.stck.pop()
        except:
            printRED('Something went wrong!')
        else:
            print(y, end=' ')
            printORNG('popped from the stack')
            self.Display()

    def Peek(self):
        print(self.stck[-1], end=' ')
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 41
FP Practical 9

```
            printBLU('is at top of the stack')
            self.Display()


    def Display(self):
            printMGNTA('Stack', end=' : ')
            print(f'{self.stck}\n')


s = Stack([1, 2, 3])
s.Push(64)
s.Peek()
s.Pop()
```

Output :