

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FP Practical 7

**Institute of Computer Technology**  
**B. Tech Computer Science and Engineering**

**Sub: (2CSE403) FUNCTIONAL PROGRAMMING**

**Practical 8**

1. A bank application corresponding to the customer of ABC bank is being developed. It takes into consideration Name of customer, its account type (saving or current), balance corresponding to the account. Account type is by default fixed from the day when user creates and account in a bank. Withdrawal & deposit are other operations which any customer would do. Also implement function called setbalance() & getbalance(). Ensure that minimum amount to be maintained is 1000/- in both savings and current. An error should be raised if such scenario occurs.

**Code:**

```
from YSL_io import *

class BnkAc:
    def __init__(self, name, ac, blnc):
        self.name = name
        self.ac = ac
        self.blnc = blnc
        self.min_blnc = 1000

        if self.blnc < self.min_blnc:
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FP Practical 7

```
        raise ValueError(f"Minimum Balance of Rs.{self.min_blnc}
required")

@property
def get_blnc(self):
    printGRN(f'\nBalance : Rs.{self.blnc}')

def set_blnc(self, setblnc):
    if setblnc < self.min_blnc:
        raise ValueError(f"Minimum Balance of Rs.{self.min_blnc}
required")
    else:
        self.blnc = setblnc
        printBLU(f"\nBalance set : Rs.{self.blnc}")

def deposit(self, amount):
    self.blnc += amount
    printGRN(f'\nDeposited : Rs.{amount}\nCurrent Balance :
Rs.{self.blnc}')

def withdraw(self, amount):
    if self.blnc - amount < self.min_blnc:
        raise ValueError(f"Minimum Balance of Rs.{self.min_blnc}
required")
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA    Batch - 41

FP Practical 7

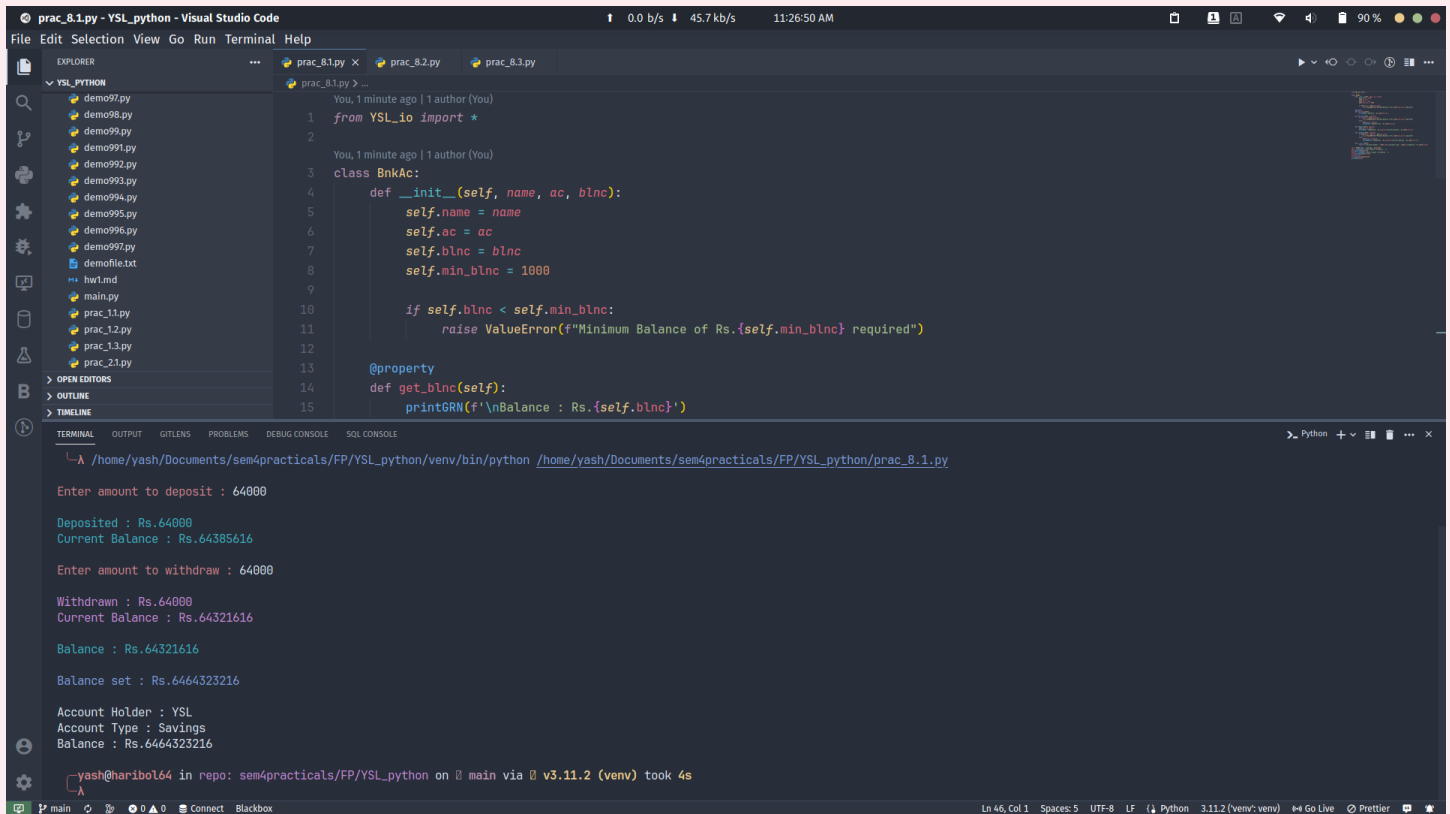
```
        else:
            self.blnc -= amount
            printMGNTA(f"\nWithdrawn : Rs.{amount}\nCurrent Balance :
Rs.{self.blnc}")

    def __str__(self):
        return f"\nAccount Holder : {self.name}\nAccount Type :
{self.ac}\nBalance : Rs.{self.blnc}"

ysl = BnkAc('YSL', 'Savings', 64321616)
dpst = inputRED('\nEnter amount to deposit : ')
ysl.deposit(int(dpst))
wthdrw = inputRED('\nEnter amount to withdraw : ')
ysl.withdraw(int(wthdrw))
ysl.get_blnc
ysl.set_blnc(6464323216)
print(str(ysl))
```

**Output :**

## FP Practical 7



Write a test program that measures the execution time of adding numbers from 1 to 1,000,000.

**Name - Yash Lakhtariya**  
**Enrollment number - 21162101012**  
**Branch - CBA    Batch - 41**  
**FP Practical 7**

**Code:**

```
from YSL_io import *

import time

class Stopwatch:
    def __init__(self):
        self.__strt = time.time()
        self.__end = 0

    def start(self):
        self.__strt = time.time()

    def stop(self):
        self.__end = time.time()

    @property
    def calculate_time(self):
        return int((self.__end - self.__strt) * 1000)

stpwtrch = Stopwatch()

total = 0

stpwtrch.start()
```

**Name - Yash Lakhtariya**  
**Enrollment number - 21162101012**  
**Branch - CBA      Batch - 41**  
**FP Practical 7**

```
for i in range(1, 1000001):
    total += i

stopwatch.stop()

printGRN(f"\nTime taken to add numbers from 1 to 1,000,000:
{stopwatch.calculate_time} ms")
```

### Output :

The image shows a Visual Studio Code interface with a dark theme. The top status bar displays 'prac\_8.2.py - YSL\_python - Visual Studio Code' and system information like '0.0 B/s', '12.3 kb/s', and '11:27:08 AM'. The Explorer sidebar on the left shows a project structure with files named 'demo97.py' through 'demo997.py', 'demofile.txt', 'hw1.md', 'main.py', and 'prac\_1.1.py' through 'prac\_2.1.py'. The main editor window is open to 'prac\_8.2.py', showing a Python script that defines a 'StopWatch' class. The class has an '\_\_init\_\_' method, a 'start' method, and a 'stop' method. The terminal at the bottom shows the execution of the script, displaying the time taken to add numbers from 1 to 1,000,000 (70 ms) and the time taken to add numbers from 1 to 1,000,000,000 (97 ms). The status bar at the bottom indicates the current file is 'main', the Python version is '3.11.2 (venv: venv)', and the Prettier extension is active.

**Name - Yash Lakhtariya**

**Enrollment number - 21162101012**

**Branch - CBA    Batch - 41**

**FP Practical 7**

3. A small module for manipulation of complex numbers is being developed for ease of research related work. Implement add() for addition, mul() for multiplication, sub() for subtraction of two complex numbers. By default, a complex number will be assigned an imaginary value of 3. Also ensure that mul() cannot be ever 0 ; so program should raise an error.

**Code:**

```
from YSL_io import *

class Complex:
    def __init__(self, real, img=3):
        self.real = real
        self.img = img

    def __add__(self, other):
        print(f"Addition of {self} and {other} : ", end=' ')
        printORNG(f'{self.real + other.real} + {self.img + other.img}i')

    def __sub__(self, other):
        print(f"Subtraction of {self} and {other} : ", end=' ')
        printORNG(f'{self.real - other.real} + {self.img - other.img}i')

    def __mul__(self, other):
        result_real = self.real * other.real - self.img * other.img
        result_img = self.real * other.img + self.img * other.real
        if result_real == 0 and result_img == 0:
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FP Practical 7

```
        raise ValueError("Multiplication result cannot be zero")
    print(f"Multiplication of {self} and {other} : ", end=' ')
    printORNG(f'{result_real} + {result_img}i')

def __str__(self):
    return f"{self.real} + {self.img}i"

r1 = int(inputGRN('\nEnter the real coefficient of a complex number 1 :
'))
i1 = int(inputGRN('Enter the imaginary coefficient of a complex number 1 :
'))
c1 = Complex(r1, i1)

r2 = int(inputGRN('Enter the real coefficient of a complex number 2 : '))
i2 = int(inputGRN('Enter the imaginary coefficient of a complex number 2 :
'))
c2 = Complex(r2, i2)

print('\n')

c1 + c2

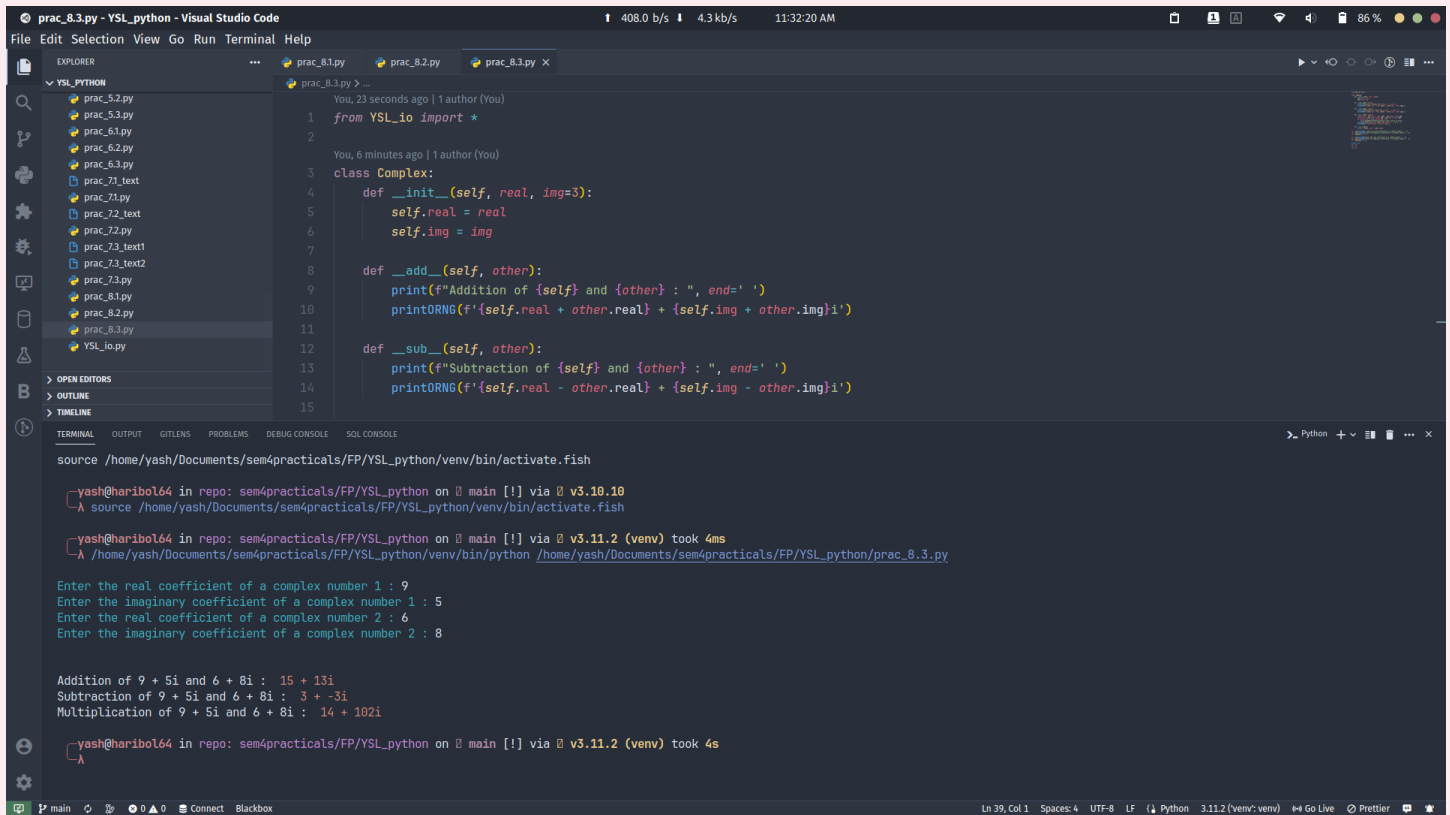
c1 - c2

c1 * c2
```



Name - Yash Lakhtariya  
Enrollment number - 21162101012  
Branch - CBA Batch - 41  
FP Practical 7

## Output :



The screenshot displays the Visual Studio Code interface with a Python file named `prac_8.3.py` open. The file contains a `Complex` class with methods for addition, subtraction, and multiplication of complex numbers. The terminal window shows the execution of the program, which prompts the user to enter the real and imaginary coefficients for two complex numbers. The output displays the results of the operations.

```
prc_8.3.py - YSL_python - Visual Studio Code
408.0 b/s | 4.3 kb/s | 11:32:20 AM

File Edit Selection View Go Run Terminal Help

EXPLORER
YSL_python
  prac_5.2.py
  prac_5.3.py
  prac_6.1.py
  prac_6.2.py
  prac_6.3.py
  prac_7.1_text
  prac_7.1.py
  prac_7.2_text
  prac_7.2.py
  prac_7.3_text1
  prac_7.3_text2
  prac_7.3.py
  prac_8.1.py
  prac_8.2.py
  prac_8.3.py
  YSL_io.py
  OPEN EDITORS
  OUTLINE
  TIMELINE

prc_8.3.py
1 from YSL_io import *
2
3 class Complex:
4     def __init__(self, real, img=3):
5         self.real = real
6         self.img = img
7
8     def __add__(self, other):
9         print(f"Addition of {self} and {other} : ", end=' ')
10        printORNG(f'{self.real + other.real} + {self.img + other.img}i')
11
12    def __sub__(self, other):
13        print(f"Subtraction of {self} and {other} : ", end=' ')
14        printORNG(f'{self.real - other.real} + {self.img - other.img}i')
15

TERMINAL
source /home/yash/Documents/sem4practicals/FP/YSL_python/venv/bin/activate.fish

yash@haribol64 in repo: sem4practicals/FP/YSL_python on  main [!] via  v3.10.10
source /home/yash/Documents/sem4practicals/FP/YSL_python/venv/bin/activate.fish

yash@haribol64 in repo: sem4practicals/FP/YSL_python on  main [!] via  v3.11.2 (venv) took 4ms
python /home/yash/Documents/sem4practicals/FP/YSL_python/prac_8.3.py

Enter the real coefficient of a complex number 1 : 9
Enter the imaginary coefficient of a complex number 1 : 5
Enter the real coefficient of a complex number 2 : 6
Enter the imaginary coefficient of a complex number 2 : 8

Addition of 9 + 5i and 6 + 8i : 15 + 13i
Subtraction of 9 + 5i and 6 + 8i : 3 + -3i
Multiplication of 9 + 5i and 6 + 8i : 14 + 102i

yash@haribol64 in repo: sem4practicals/FP/YSL_python on  main [!] via  v3.11.2 (venv) took 4s
```