

PYTHON JOURNAL

Name: **Rashi Sawardekar** Roll No: **31010924093** FYIT- B

PRACTICAL 1

[all files](#)

Variables

Aim:

Declare Variables and implement Implicit and Explicit Typecasting, understand errors.

Source Code:

```
print("IMPLICIT TYPE Casting")
a=19
print(type(a))
b=19.5
print(type(b))
c="raya"
print(type(c))
d=True
print(type(d))
e=3+5j
print(type(e))
print("EXPLICIT TYPE CASTING")
R="21"
print(type(R))
print(chr(89))
J=int(R)
print(type(J))
M=float(R)
print(M)
```

```
print(type(M))
Y=str(R)
print(Y)
print(type(Y))
f=3.143
g=int(f)
print(g)
alphabet=ord('Y')
number=ord('1')
print("Ascii Value of Character Y is:" ,alphabet)
print("Ascii Value of character digit 1 is:" ,number)
dec=1921
print("The Value of ",dec, "in other number system")
print(bin(dec),"in binary number")
print(oct(dec),"in octal")
print(hex(dec),"in hexadecimal")
```

Output:

1.py - C:\Users\yashraj\Desktop\RASHI\1.py (3.12.9)

File Edit Format Run Options Window Help

```
print("IMPLICIT TYPE Casting")
a=19
print(type(a))
b=19.5
print(type(b))
c="raya"
print(type(c))
d=True
print(type(d))
e=3+5j
print(type(e))
print("EXPLICIT TYPE CASTING")
R="21"
print(type(R))
print(chr(89))
J=int(R)
print(type(J))
M=float(R)
print(M)
print(type(M))
Y=str(R)
print(Y)
print(type(Y))
f=3.143
g=int(f)
print(g)
alphabet=ord('Y')
number=ord('1')
print("Ascii Value of Character Y is:" ,alphabet)
print("Ascii Value of character digit 1 is:" ,number)
dec=1921
print("The Value of ",dec, "in other number system")
print(bin(dec),"in binary number")
print(oct(dec),"in octal")
print(hex(dec),"in hexadecimal")
```

```
1.py - C:\Users\yashraj\Desktop\RASHI\1.py (3.12.9)
File Edit Shell Debug Options Window Help
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb  4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\yashraj\Desktop\RASHI\1.py =====
IMPLICIT TYPE Casting
<class 'int'>
<class 'float'>
<class 'str'>
<class 'bool'>
<class 'complex'>
EXPLICIT TYPE CASTING
<class 'str'>
Y
<class 'int'>
21.0
<class 'float'>
21
<class 'str'>
3
Ascii Value of Character Y is: 89
Ascii Value of character digit 1 is: 49
The Value of 1921 in other number system
0b11110000001 in binary number
0o3601 in octal
0x781 in hexadecimal
>>>
```

Conclusion:

The code is executed successfully.

PRACTICAL 2

Conditions

A.

Aim:

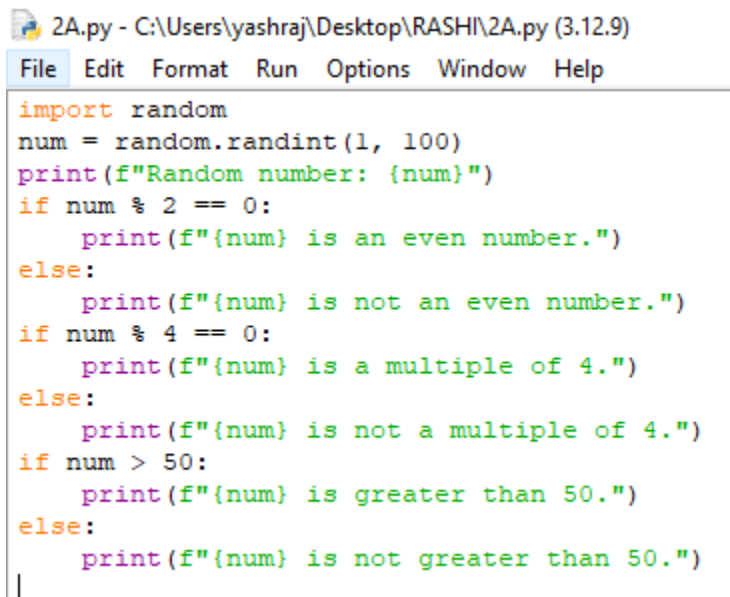
Write a Python program that:

- a. Declares an int type variable num.
- b. Assign a random value to the num between 1-100.
- c. Program should check and displays the following messages.
if:
 - i. The num is an even number
 - ii. The num is multiple of 4
 - iii. The num is greater than 50

Source Code:

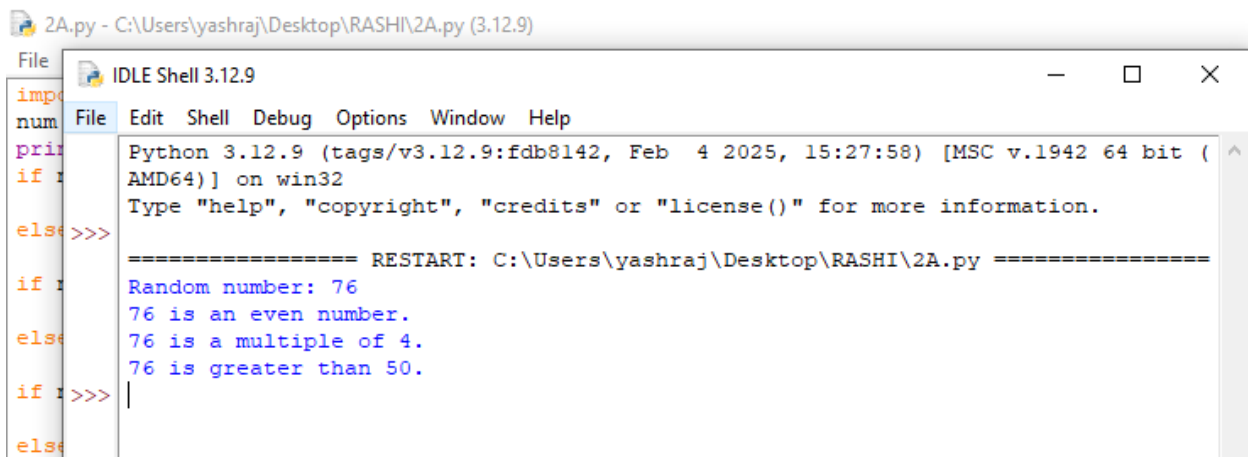
```
import random
num = random.randint(1, 100)
print(f"Random number: {num}")
if num % 2 == 0:
    print(f"{num} is an even number.")
else:
    print(f"{num} is not an even number.")
if num % 4 == 0:
    print(f"{num} is a multiple of 4.")
else:
    print(f"{num} is not a multiple of 4.")
if num > 50:
    print(f"{num} is greater than 50.")
else:
    print(f"{num} is not greater than 50.")
```

Output:



```
2A.py - C:\Users\yashraj\Desktop\RASHI\2A.py (3.12.9)
File Edit Format Run Options Window Help

import random
num = random.randint(1, 100)
print(f"Random number: {num}")
if num % 2 == 0:
    print(f"{num} is an even number.")
else:
    print(f"{num} is not an even number.")
if num % 4 == 0:
    print(f"{num} is a multiple of 4.")
else:
    print(f"{num} is not a multiple of 4.")
if num > 50:
    print(f"{num} is greater than 50.")
else:
    print(f"{num} is not greater than 50.")
|
```



```
2A.py - C:\Users\yashraj\Desktop\RASHI\2A.py (3.12.9)
File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:\Users\yashraj\Desktop\RASHI\2A.py =====
Random number: 76
76 is an even number.
76 is a multiple of 4.
76 is greater than 50.
if >>>
else
```

Conclusion:

The code is executed successfully.

B.**Aim:**

Write a program in python to find whether a string is PALINDROME or not.

Source Code:

```
def is_palindrome(s):  
    return s == s[::-1]  
  
string = input("Enter a string: ")  
  
if is_palindrome(string):  
    print(f"{string}' is a palindrome.")  
else:  
    print(f"{string}' is not a palindrome.")
```

Output:

2B.py - C:\Users\yashraj\Desktop\RASHI\2B.py (3.12.9)

File Edit Format Run Options Window Help

```
def is_palindrome(s):  
    return s == s[::-1]  
  
string = input("Enter a string: ")  
  
if is_palindrome(string):  
    print(f"'{string}' is a palindrome.")  
else:  
    print(f"'{string}' is not a palindrome.")
```

2B.py - C:\Users\yashraj\Desktop\RASHI\2B.py (3.12.9)

File Edit Format Run Options Window Help

```
def is_pal  
    return
```

```
string = i
```

```
if is_pali
```

```
    print(>>>
```

```
else:
```

```
    print(>>>
```

```
>>>
```

```
>>>
```

```
>>>
```

```
>>>
```

```
>>>
```

IDLE Shell 3.12.9

File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:\Users\yashraj\Desktop\RASHI\2B.py =====

Enter a string: mom

'mom' is a palindrome.

>>>

===== RESTART: C:\Users\yashraj\Desktop\RASHI\2B.py =====

Enter a string: rash

'rash' is not a palindrome.

>>>

Conclusion:

The code is executed successfully.

PRACTICAL 3

Loops

A.

Aim:

Write Processing code that declares an integer variable called limit and assigns a random value between 1 and 50. Your program should, then, calculate the sum of all the squares of the integers from 1 to limit (inclusive) using a loop. The result should be assigned to a variable named result. Print the value of both variables limit and result to check the correctness of your program.

Source Code:

```
import random
```

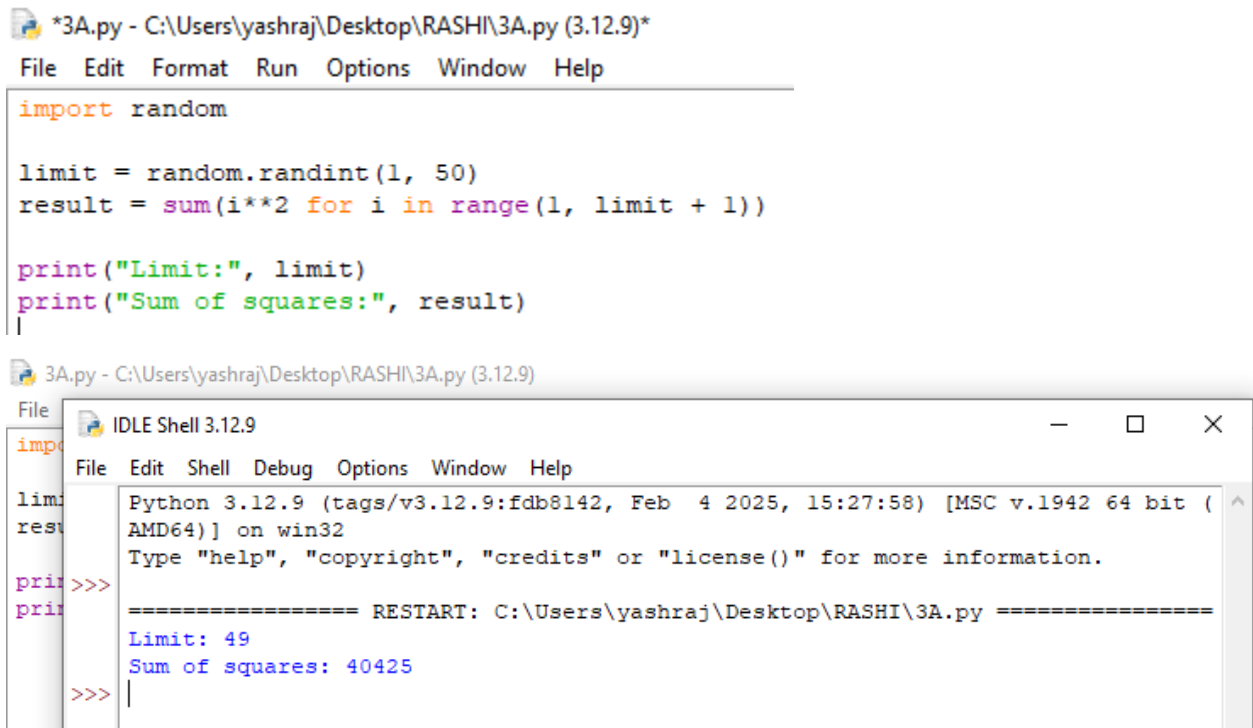
```
limit = random.randint(1, 50)
```

```
result = sum(i**2 for i in range(1, limit + 1))
```

```
print("Limit:", limit)
```

```
print("Sum of squares:", result)
```

Output:



The screenshot displays the Python IDLE environment. The top window, titled '*3A.py - C:\Users\yashraj\Desktop\RASHI\3A.py (3.12.9)*', contains the following Python code:

```
import random

limit = random.randint(1, 50)
result = sum(i**2 for i in range(1, limit + 1))

print("Limit:", limit)
print("Sum of squares:", result)
```

The bottom window, titled 'IDLE Shell 3.12.9', shows the execution output. It includes the Python version and architecture information, followed by a restart message and the program's output:

```
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:\Users\yashraj\Desktop\RASHI\3A.py =====
Limit: 49
Sum of squares: 40425
>>>
```

Conclusion:

The code is executed successfully.

B.

Aim:

A common mathematical problem is to find a solution of arithmetic series. An arithmetic series is the sum of the terms of an arithmetic sequence. Write a Python program to find the answer A of the following arithmetic series.

$$A = a_0 + \sum_{n=5}^{30} (a \cos \frac{n\pi x}{L} + b \sin \frac{n\pi x}{L})$$

Where $a_0 = 7.86$, $a = 16.67$, $b = 76.667$, $L = 99$.

Source Code:

```
import math

a0,a,b,L,n = 7.86,16.67,76.667,99,5

A = 0

x=int(input("Enter an positive integer:"))

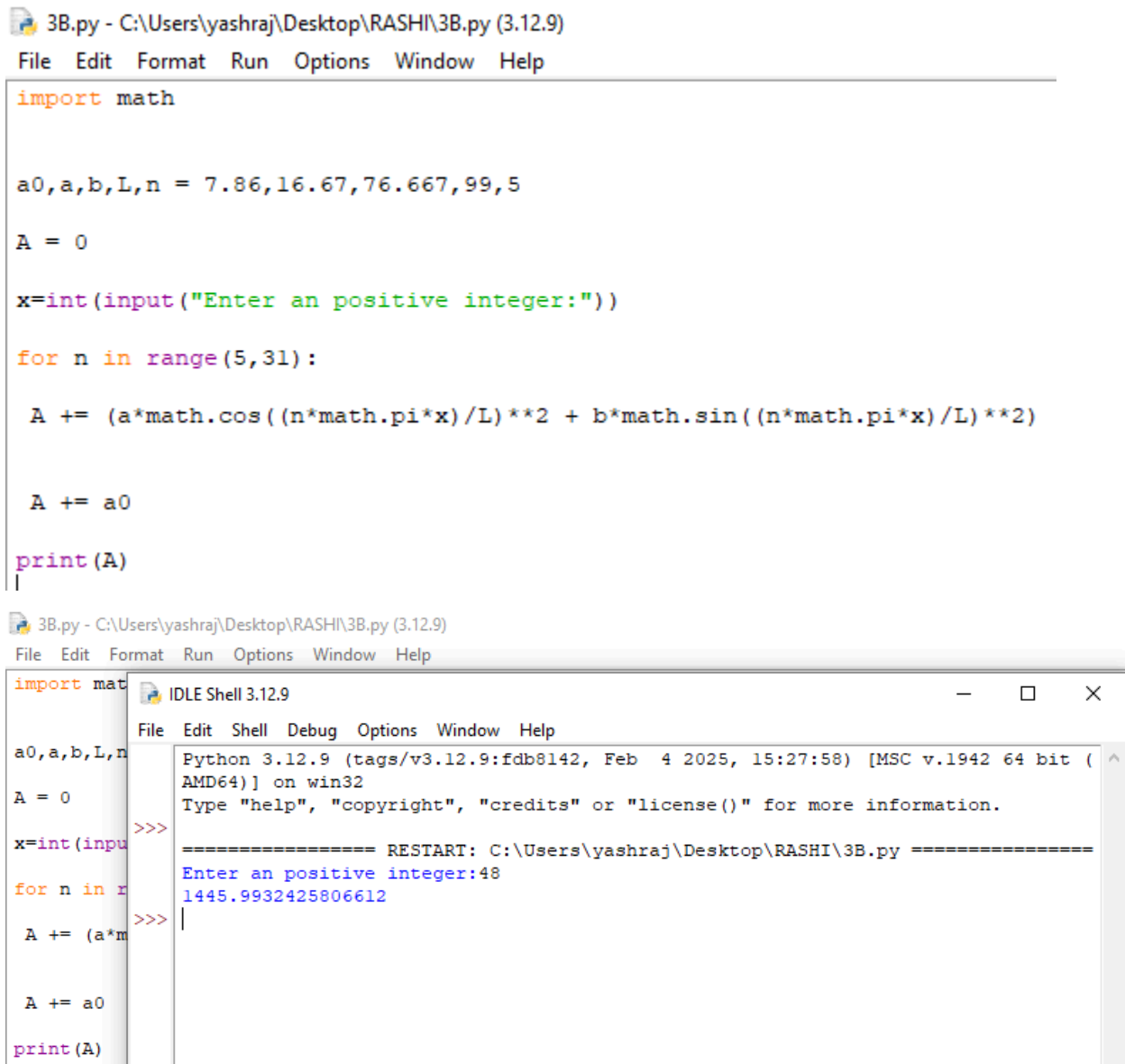
for n in range(5,31):

    A += (a*math.cos((n*math.pi*x)/L)**2 +
b*math.sin((n*math.pi*x)/L)**2)

A += a0

print(A)
```

Output:



```
3B.py - C:\Users\yashraj\Desktop\RASHI\3B.py (3.12.9)
File Edit Format Run Options Window Help

import math

a0,a,b,L,n = 7.86,16.67,76.667,99,5

A = 0

x=int(input("Enter an positive integer:"))

for n in range(5,31):

    A += (a*math.cos((n*math.pi*x)/L)**2 + b*math.sin((n*math.pi*x)/L)**2)

    A += a0

print(A)
|

3B.py - C:\Users\yashraj\Desktop\RASHI\3B.py (3.12.9)
File Edit Format Run Options Window Help

import mat
a0,a,b,L,n
A = 0
x=int(inp
for n in r
    A += (a*m
    A += a0
print(A)
```

IDLE Shell 3.12.9

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>> ===== RESTART: C:\Users\yashraj\Desktop\RASHI\3B.py =====
Enter an positive integer:48
1445.9932425806612
>>> |

Conclusion:

The code is executed successfully.

C.

Aim:

Write a program to declare an integer type variable number.
Display the cube of all integers from 1 to number. For example, if a number is 5, then expected output is as:

Number is : 1 and cube of the 1 is :1

Number is : 2 and cube of the 2 is :8

Number is : 3 and cube of the 3 is :27

Number is : 4 and cube of the 4 is :64

Number is : 5 and cube of the 5 is :125

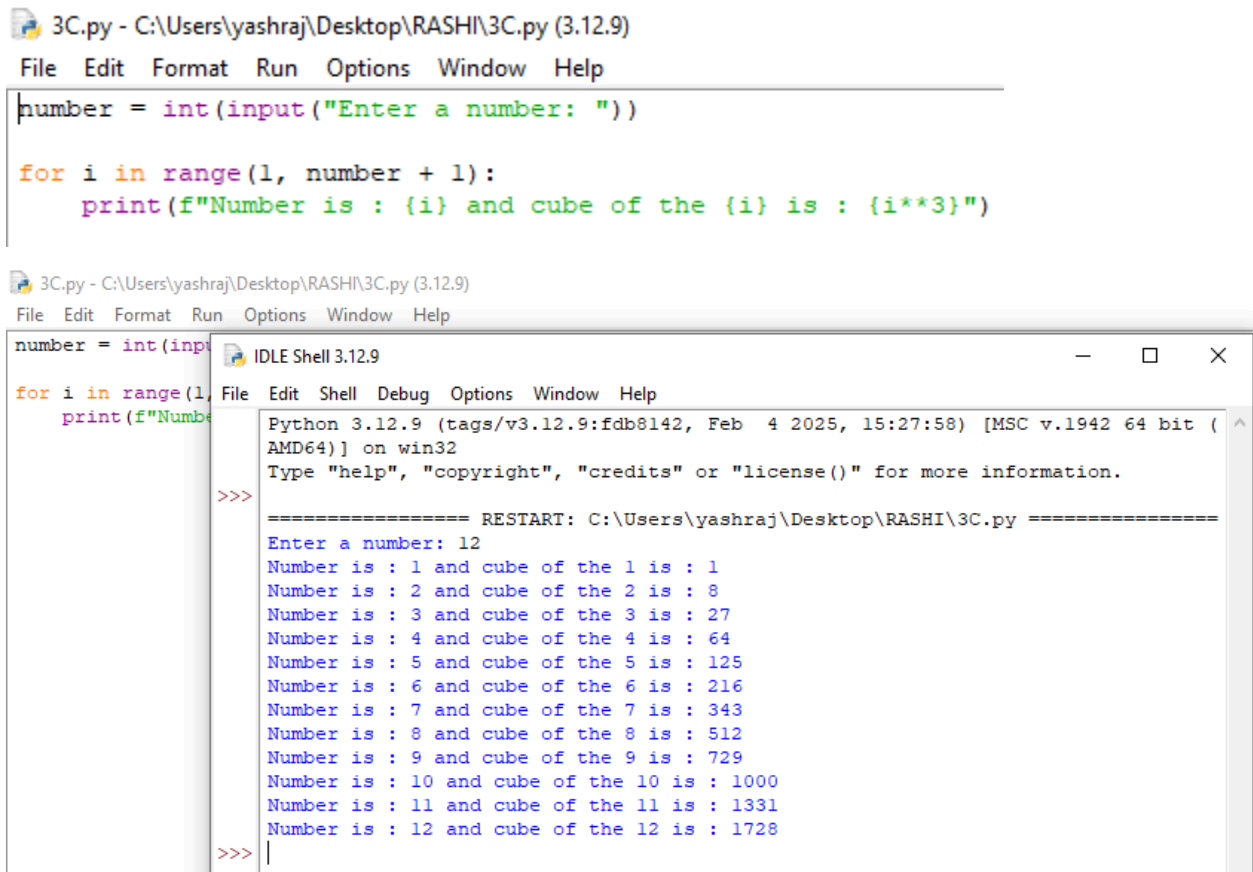
Source Code:

```
number = int(input("Enter a number: "))
```

```
for i in range(1, number + 1):
```

```
    print(f"Number is : {i} and cube of the {i} is : {i**3}")
```

Output:



The image shows two screenshots of a Python 3.12.9 IDE. The top screenshot displays the source code for a program that calculates the cube of numbers from 1 to a user-defined number. The bottom screenshot shows the same code being executed in the IDLE Shell, with the output displayed in the console window.

Source Code:

```
3C.py - C:\Users\yashraj\Desktop\RASHI\3C.py (3.12.9)
File Edit Format Run Options Window Help

number = int(input("Enter a number: "))

for i in range(1, number + 1):
    print(f"Number is : {i} and cube of the {i} is : {i**3}")
```

Execution Output:

```
IDLE Shell 3.12.9
Python 3.12.9 (tags/v3.12.9:fd8b142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\yashraj\Desktop\RASHI\3C.py =====
Enter a number: 12
Number is : 1 and cube of the 1 is : 1
Number is : 2 and cube of the 2 is : 8
Number is : 3 and cube of the 3 is : 27
Number is : 4 and cube of the 4 is : 64
Number is : 5 and cube of the 5 is : 125
Number is : 6 and cube of the 6 is : 216
Number is : 7 and cube of the 7 is : 343
Number is : 8 and cube of the 8 is : 512
Number is : 9 and cube of the 9 is : 729
Number is : 10 and cube of the 10 is : 1000
Number is : 11 and cube of the 11 is : 1331
Number is : 12 and cube of the 12 is : 1728
>>>
```

Conclusion:

The code is executed successfully.

D.

Aim:

Trace how many times the statement will be executed in each of the following the following loops? Afterwards, replace the statement with `println(j)` and type the statements to check how many times the value of the variable `j` was printed. Note: convert into python from java.

1. `for(int i=1;i<=10;i++)`

`for (int j = 1; j <= 10; j++)`
`statement;`

2. `for(int i=1;i<=10;i++)`

`for (int j = 1; j <= i; j++)`
`statement;`

3. `for(int i=1;i<=5;i++)`

`for (int j = 10; j > i; j--)`
`statement`

Source Code:


```
count = 0
for i in range(1, 11):
    for j in range(1, 11):
        print(j)
        count += 1
print("Total executions:", count)
```

```
count = 0
for i in range(1, 11):
    for j in range(1, i + 1):
```

```
    print(j)
    count += 1
print("Total executions:", count)
```

```
count = 0
for i in range(1, 6):
    for j in range(10, i, -1):
        print(j)
        count += 1
print("Total executions:", count)
```

Output:

 3D.py - C:/Users/yashraj/Desktop/RASHI/3D.py (3.12.9)

File Edit Format Run Options Window Help

```
count = 0
for i in range(1, 11):
    for j in range(1, 11):
        print(j)
        count += 1
print("Total executions:", count)
```

```
count = 0
for i in range(1, 11):
    for j in range(1, i + 1):
        print(j)
        count += 1
print("Total executions:", count)
```

```
count = 0
for i in range(1, 6):
    for j in range(10, i, -1):
        print(j)
        count += 1
print("Total executions:", count)
```

|

File Edit Format Run Options Window Help

```
count = 0
for i in range(10):
    for j in range(10):
```

```
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
```

```
count = 0
for i in range(1, 10):
    print(i)
    count += 1
    if count == 5:
        break
    else:
        continue
print("Loop ended")
```

```
for i in r: 1
    for j: 2
        pr: 3
        co: 4
print("Tot: 5
count = 0 6
for i in r: 7
    for j: 8
        pr: 9
        co: 10
print("Tot: 1
2
```

1
2
3
4
5
6
7
8
9
10
1
2
3
4
5
6
7
8
9
10
1
2
3
4
5
6
7
8
9
10
1
2
3
4
5

File Edit Format Run Options Window Help

File Edit Format Run Options Window Help

count = 0

```
for i in range(1, 10):
    for j in range(1, 10):
```

File Edit Shell Debug Options Window Help

pr 4

```
print("Tot
```

7

```
count = 0
for i in r
```

```
for j
```

pr	1
co	2

```
print("Tot
```

```
count = 0
```

```
for i in range(1, 10):
    for j in range(1, 10):
```

```
for j in range(1, 7):
    print(j)
```

CO 9

```
print("100")
```

1
2

	3	
	4	

5

	6	
	7	

8

9
10

	2	
	3	

4

5	
6	

8	
9	

10

	1	
	2	

3D.py - C:/Users/yashraj/Desktop/RASHI/3D.py (3.12.9)

File Edit Format Run Options Window Help

```
count = 0
for i in r
    for j
        pr
        co
print("Tot

count = 0
for i in r
    for j
        pr
        co
print("Tot

count = 0
for i in r
    for j
        pr
        co
print("Tot
```

IDLE Shell 3.12.9

File Edit Shell Debug Options Window Help

```
1
2
3
1
2
3
4
1
2
3
4
5
1
2
3
4
5
6
1
2
3
4
5
6
7
1
2
3
4
5
6
7
8
1
2
3
4
5
6
```


The screenshot shows the Python IDLE Shell 3.12.9 interface. The main window displays the following code:

```
count = 0
for i in range(10):
    for j in range(10):
        print("Total executions: 55")
        count = 0
    for i in range(10):
        for j in range(10):
            print("Total executions: 35")
```

The output in the shell window shows the execution of the code, with the following lines visible:

```
9
10
Total executions: 55
10
9
8
7
6
5
4
3
2
10
9
8
7
6
5
4
10
9
8
7
6
5
10
9
8
7
6
Total executions: 35
```

Conclusion:

The code is executed successfully.

E.

Aim:

Write a loop that output the following pattern of numbers.

a. 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5 6 6 6 6 6 6

b. 1 1 1 1 1 1 2 2 2 2 2 3 3 3 3 4 4 4 5 5 6

c. 1 1 2 1 2 3 1 2 3 4 1 2 3 4 5 1 2 3 4 5 6

Source Code:

```
print("Pattern a:")
for i in range(1, 7):
    for j in range(i):
        print(i, end=" ")
    print()
```

```
print("Pattern b:")
for i in range(1, 7):
    for j in range(6 - i + 1):
        print(i, end=" ")
    print()
```

```
print("Pattern c:")
for i in range(1, 7):
    for j in range(1, i + 1):
        print(j, end=" ")
    print()
```

Output:



```
3E.py - C:/Users/yashraj/Desktop/RASHI/3E.py (3.12.9)
File Edit Format Run Options Window Help

print("Pattern a:")
for i in range(1, 7):
    for j in range(i):
        print(i, end=" ")
    print()

print("Pattern b:")
for i in range(1, 7):
    for j in range(6 - i + 1):
        print(i, end=" ")
    print()

print("Pattern c:")
for i in range(1, 7):
    for j in range(1, i + 1):
        print(j, end=" ")
    print()

3E.py - C:/Users/yashraj/Desktop/RASHI/3E.py (3.12.9)
File Edit Format Run Options Window Help

print("Pattern a:")
for i in range(1, 7):
    for j in range(i):
        print(i, end=" ")
    print()

print("Pattern b:")
for i in range(1, 7):
    for j in range(6 - i + 1):
        print(i, end=" ")
    print()

print("Pattern c:")
for i in range(1, 7):
    for j in range(1, i + 1):
        print(j, end=" ")
    print()

IDLE Shell 3.12.9
File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/3E.py =====
Pattern a:
1 2 2 3 3 3 4 4 4 4 5 5 5 5 6 6 6 6 6 6
Pattern b:
1 1 1 1 1 1 2 2 2 2 2 3 3 3 4 4 4 5 5 6
Pattern c:
1 1 2 1 2 3 1 2 3 4 1 2 3 4 5 1 2 3 4 5 6
```

Conclusion:

The code is executed successfully.

PRACTICAL 4

Functions

A.

Aim:

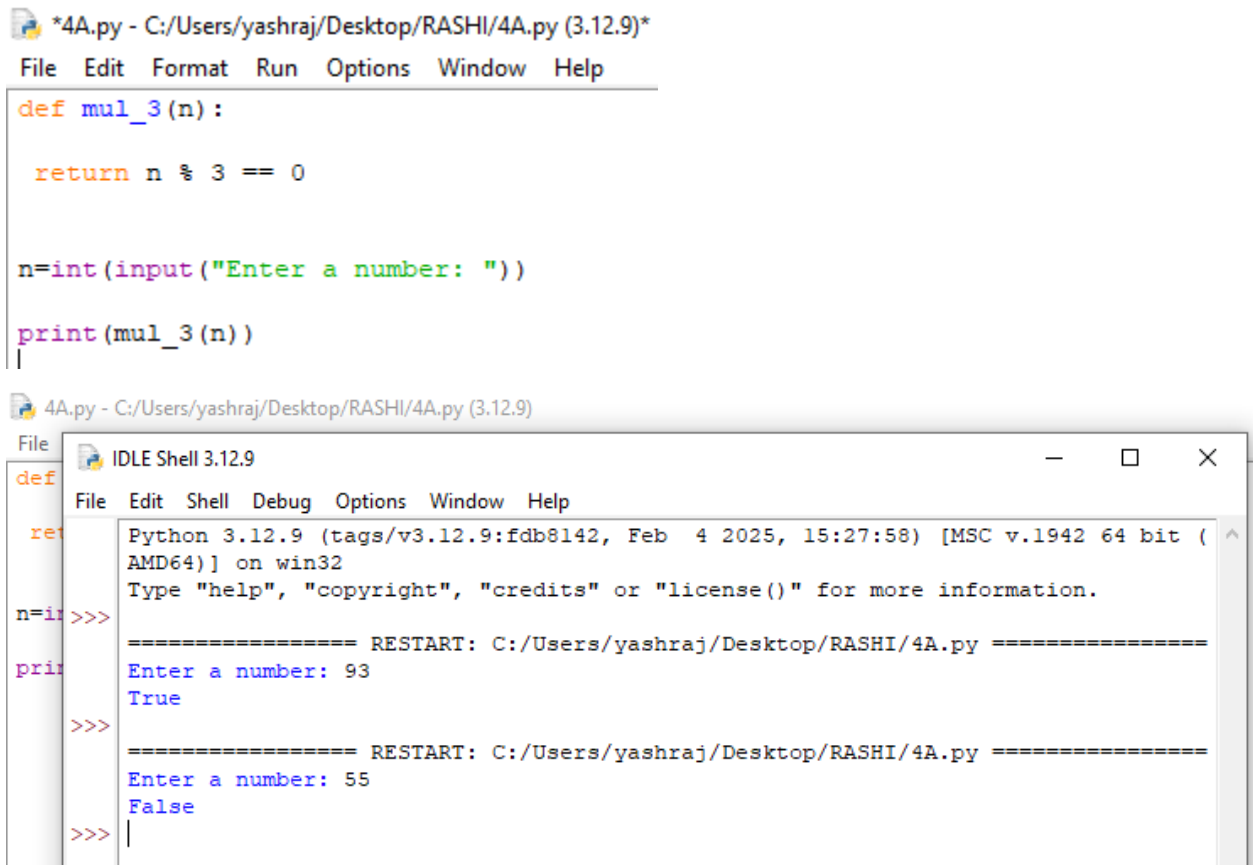
User Defined function

Write a user defined function when passed a value of type int, returns true if the value is multiple of 3 and returns false otherwise. (Multiples of 3 are 3, 6, 9, 12)

Source Code:

```
def mul_3(n):  
    return n % 3 == 0  
  
n=int(input("Enter a number: "))  
print(mul_3(n))
```

Output:



```
*4A.py - C:/Users/yashraj/Desktop/RASHI/4A.py (3.12.9)*
File Edit Format Run Options Window Help

def mul_3(n):
    return n % 3 == 0

n=int(input("Enter a number: "))
print(mul_3(n))
|

4A.py - C:/Users/yashraj/Desktop/RASHI/4A.py (3.12.9)
File Edit Shell Debug Options Window Help
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/4A.py =====
Enter a number: 93
True
>>>

===== RESTART: C:/Users/yashraj/Desktop/RASHI/4A.py =====
Enter a number: 55
False
>>>
```

Conclusion:

The code is executed successfully.

B.

Aim:

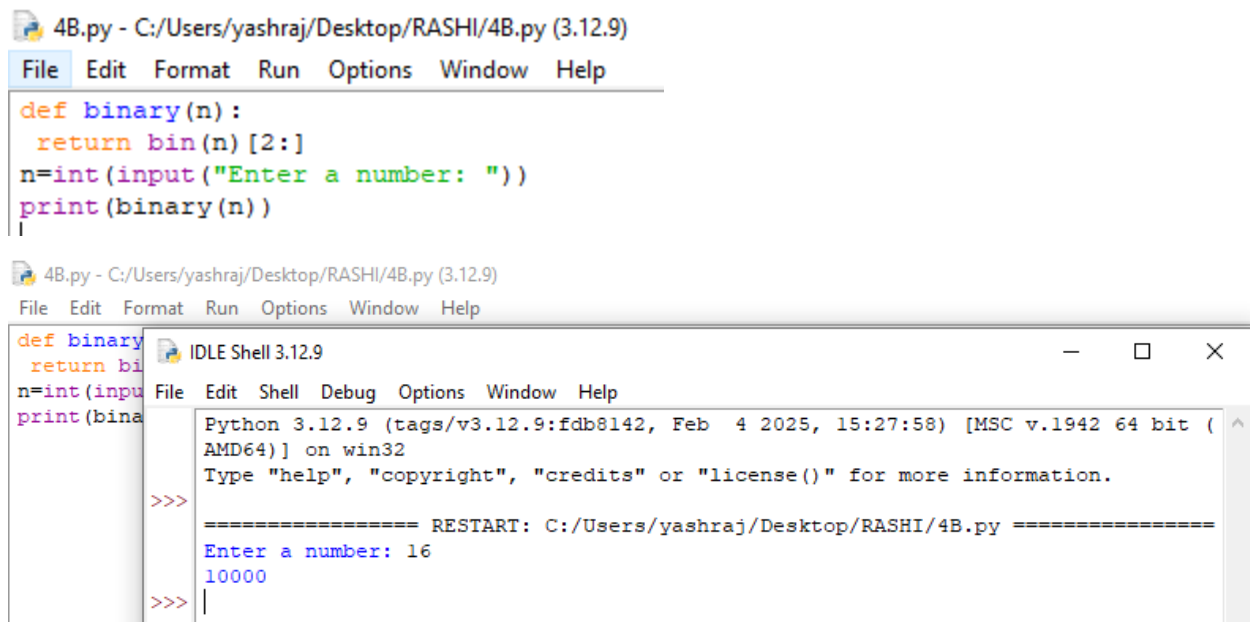
Keyword as Argument

Write a function that when passed an int type number as an argument, returns the binary equivalent as a string of 0s and 1s. For example, if we pass the value 13 to that function, should return "1101".

Source Code:

```
def binary(n):  
    return bin(n)[2:]  
  
n=int(input("Enter a number: "))  
  
print(binary(n))
```

Output:



The screenshot displays the Python IDLE 3.12.9 environment. The top window shows the source code for a function named 'binary' that takes an integer 'n' as an argument and returns its binary representation as a string. The code prompts the user to 'Enter a number: ' and prints the result. The bottom window, titled 'IDLE Shell 3.12.9', shows the execution of the code. It displays the Python version and system information, followed by a restart message. The user enters '16', and the program outputs '10000'.

```
4B.py - C:/Users/yashraj/Desktop/RASHI/4B.py (3.12.9)  
File Edit Format Run Options Window Help  
def binary(n):  
    return bin(n)[2:]  
n=int(input("Enter a number: "))  
print(binary(n))  
|  
  
4B.py - C:/Users/yashraj/Desktop/RASHI/4B.py (3.12.9)  
File Edit Format Run Options Window Help  
def binary  
    return bi  
n=int(inp  
print(bina  
IDLE Shell 3.12.9  
File Edit Shell Debug Options Window Help  
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>>  
===== RESTART: C:/Users/yashraj/Desktop/RASHI/4B.py =====  
Enter a number: 16  
10000  
>>>|
```

Conclusion:

The code is executed successfully.

C.

Aim:

Boolean Function

Write a Boolean function that when passed a number should return whether true or false after checking if that number is an "Armstrong" number or not. An integer number is called Armstrong number if sum of the cubes of its digits is equal to the number itself. For example, 371 is an Armstrong number since $3^3 + 7^3 + 1^3 = 371$.

Note: 153, 370, 371, 407 are all Armstrong numbers. You can use these values to check correctness of your function.

Source Code:

```
def armst(n):  
    return n==sum(int(digit)**3 for digit in str(n))  
  
n=int(input("Enter a number: "))  
  
print(armst(n))
```

Output:

4C.py - C:/Users/yashraj/Desktop/RASHI/4C.py (3.12.9)

File Edit Format Run Options Window Help

```
def armst(n):  
    return n==sum(int(digit)**3 for digit in str(n))  
n=int(input("Enter a number: "))  
print(armst(n))  
|
```

4C.py - C:/Users/yashraj/Desktop/RASHI/4C.py (3.12.9)

File Edit Format Run Options Window Help

```
def armst(n):  
    return n==sum(int(digit)**3 for digit in str(n))  
n=int(input("Enter a number: "))  
print(armst(n))
```

IDLE Shell 3.12.9

File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>> ===== RESTART: C:/Users/yashraj/Desktop/RASHI/4C.py =====
Enter a number: 41

False

>>> ===== RESTART: C:/Users/yashraj/Desktop/RASHI/4C.py =====
Enter a number: 153

True

>>> |

Conclusion:

The code is executed successfully.

D.

Aim:

Recursive Function

Write a Python Recursive function to calculate the factorial of a number (a non-negative integer). The function accepts the number as an argument

Source Code:

```
def fact(n):  
    if n==0:  
        return 1  
    return n*fact(n-1)  
n=int(input("Enter a number: "))  
print(fact(n))
```

Output:

4D.py - C:/Users/yashraj/Desktop/RASHI/4D.py (3.12.9)

File Edit Format Run Options Window Help

```
def fact(n):  
    if n==0:  
        return 1  
    return n*fact(n-1)  
n=int(input("Enter a number: "))  
print(fact(n))  
|
```

4D.py - C:/Users/yashraj/Desktop/RASHI/4D.py (3.12.9)

File Edit Format Run Options Window Help

def fact(n)
if n==0:
return 1
return n*
n=int(inp
print(fact

IDLE Shell 3.12.9

File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

==== RESTART: C:/Users/yashraj/Desktop/RASHI/4D.py ====

Enter a number: 6

720

>>> |

Conclusion:

The code is executed successfully.

PRACTICAL 5

Strings

A.

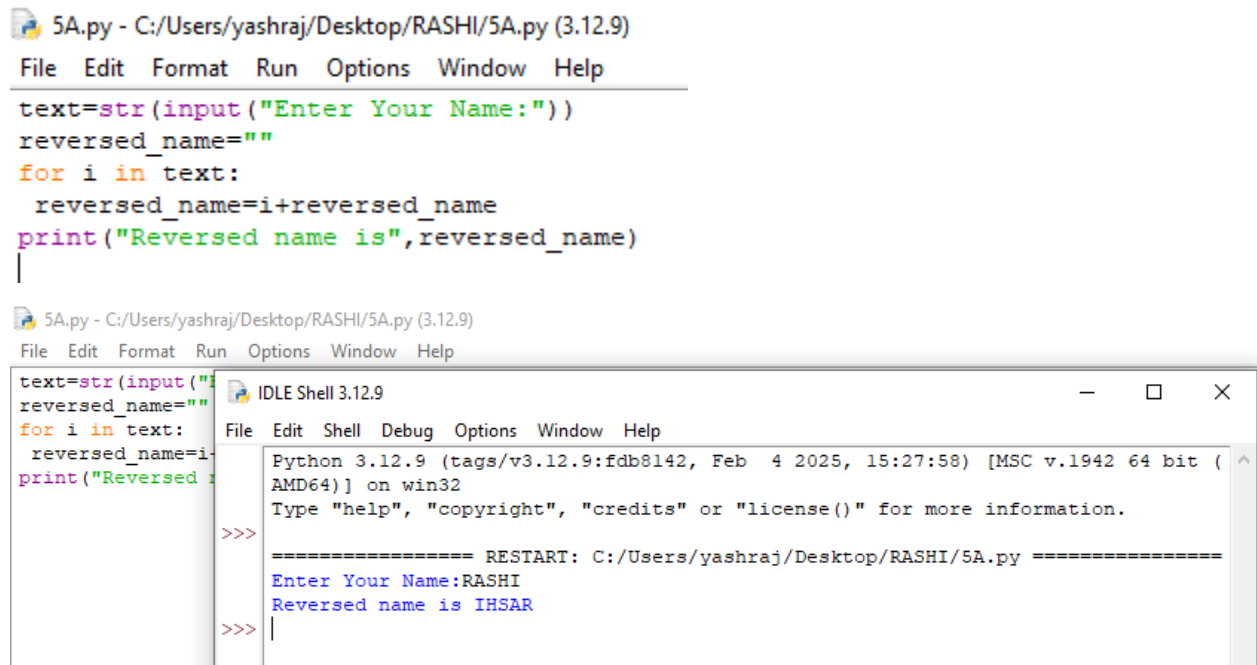
Aim:

Write a program that declares a String variable called name and initializes it with your full name. Print the value of that variable in reverse order using a loop.

Source Code:

```
text=str(input("Enter Your Name:"))
reversed_name=""
for i in text:
    reversed_name=i+reversed_name
print("Reversed name is",reversed_name)
```


Output:



The image shows two windows from the Python IDLE environment. The top window is the editor for '5A.py', showing the following code:

```
5A.py - C:/Users/yashraj/Desktop/RASHI/5A.py (3.12.9)
File Edit Format Run Options Window Help
text=str(input("Enter Your Name:"))
reversed_name=""
for i in text:
    reversed_name=i+reversed_name
print("Reversed name is",reversed_name)
|
```

The bottom window is the 'IDLE Shell 3.12.9' interpreter. It displays the program's output after execution:

```
IDLE Shell 3.12.9
File Edit Shell Debug Options Window Help
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>>
===== RESTART: C:/Users/yashraj/Desktop/RASHI/5A.py =====
Enter Your Name:RASHI
Reversed name is IHSAR
>>> |
```

Conclusion:

The code is executed successfully.

B.

Aim:

Write a program to count the total number of words in a String.
For example, the string "Welcome to SOMAIYA" has 3 words.

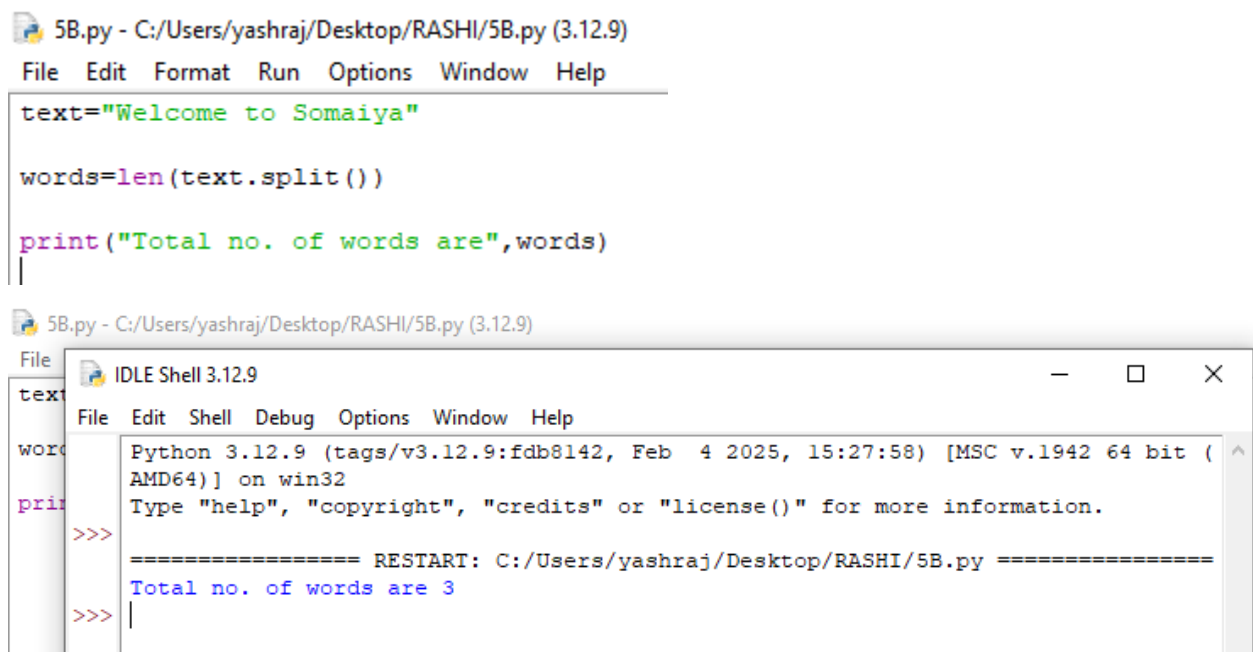
Source Code:

```
text="Welcome to Somaiya"

words=len(text.split())

print("Total no. of words are",words)
```

Output:



The screenshot displays two windows from a Python IDE. The top window, titled '5B.py - C:/Users/yashraj/Desktop/RASHI/5B.py (3.12.9)', shows the source code: `text="Welcome to Somaiya"`, `words=len(text.split())`, and `print("Total no. of words are",words)`. The bottom window, titled 'IDLE Shell 3.12.9', shows the execution output: `Total no. of words are 3`. The shell window also displays the Python version (3.12.9) and the file path.

Conclusion:

The code is executed successfully.

C.

Aim:

Write a program that declares a String variable called password and initializes it with some value. The program should then check if the password meets the following criteria or not. The program should print an error message if the password does not meet the criteria.

Criteria-

- i. The length of password should be at least 8 characters
- ii. The password should have at least 1 digit
- iii. The password should have at least 1 capital letter.

Source Code:

```
def check_pass(password):  
    if len(password) < 8:  
        print("Error! Password must be at least 8 characters long.")  
        return False  
    if not any(char.isdigit() for char in password):  
        print("Error! Password must contain at least one digit.")  
        return False  
    if not any(char.isupper() for char in password):  
        print("Error! Password must contain at least one uppercase  
letter.")  
        return False
```

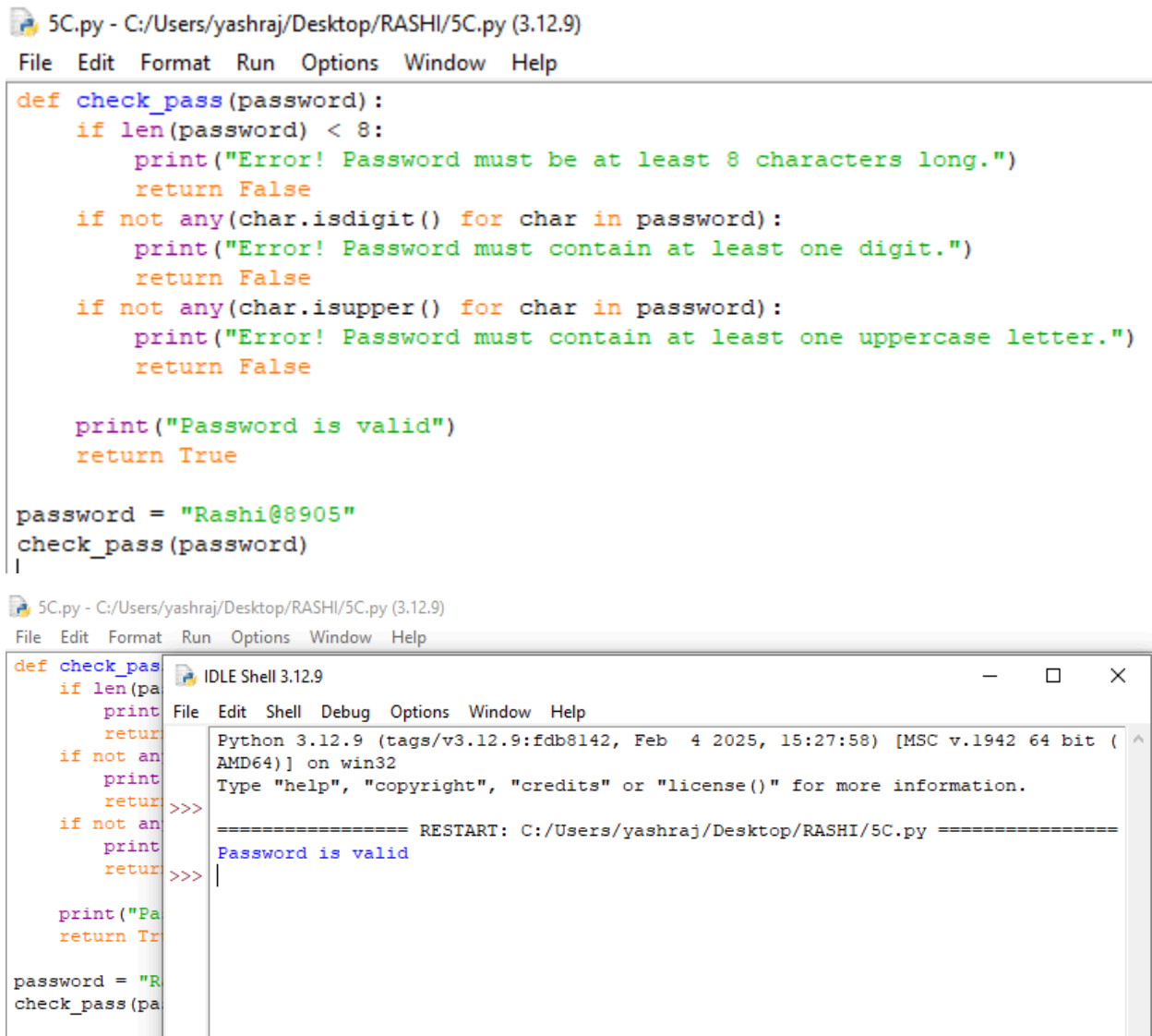
```
print("Password is valid")
```

```
return True
```

```
password = "Rashi@8905"
```

```
check_pass(password)
```

Output:



```
5C.py - C:/Users/yashraj/Desktop/RASHI/5C.py (3.12.9)
File Edit Format Run Options Window Help

def check_pass(password):
    if len(password) < 8:
        print("Error! Password must be at least 8 characters long.")
        return False
    if not any(char.isdigit() for char in password):
        print("Error! Password must contain at least one digit.")
        return False
    if not any(char.isupper() for char in password):
        print("Error! Password must contain at least one uppercase letter.")
        return False

    print("Password is valid")
    return True

password = "Rashi@8905"
check_pass(password)
|

5C.py - C:/Users/yashraj/Desktop/RASHI/5C.py (3.12.9)
File Edit Format Run Options Window Help

def check_pas
if len(pa
print
return
if not an
print
return
if not an
print
return
print("Pa
return Tr

password = "R
check_pass(pa

IDLE Shell 3.12.9
File Edit Shell Debug Options Window Help
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/yashraj/Desktop/RASHI/5C.py =====
Password is valid
>>> |
```

Conclusion:

The code is executed successfully.

D.

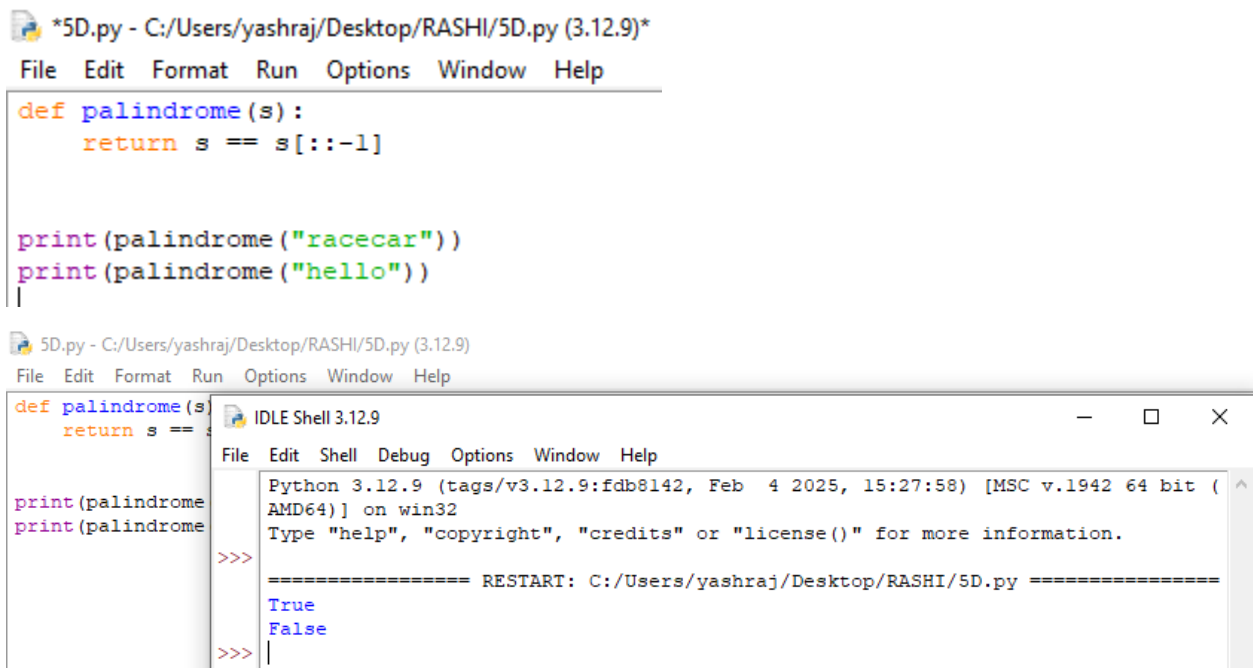
Aim:

Compute a function palindrome that when passed a String value returns true if the value is palindrome and false otherwise.

Source Code:

```
def palindrome(s):  
    return s == s[::-1]  
  
print(palindrome("racecar"))  
print(palindrome("hello"))
```

Output:



The screenshot displays two windows from the Python IDLE environment. The top window, titled '*5D.py - C:/Users/yashraj/Desktop/RASHI/5D.py (3.12.9)*', shows the source code for the palindrome function and its execution. The code defines a function 'palindrome(s)' that returns 's == s[::-1]', and then prints the results for 'racecar' and 'hello'. The bottom window, titled 'IDLE Shell 3.12.9', shows the output of the code. It displays the Python version (3.12.9), the operating system (win32), and the results of the function calls: 'True' for 'racecar' and 'False' for 'hello'.

```
*5D.py - C:/Users/yashraj/Desktop/RASHI/5D.py (3.12.9)*  
File Edit Format Run Options Window Help  
def palindrome(s):  
    return s == s[::-1]  
  
print(palindrome("racecar"))  
print(palindrome("hello"))  
|  
  
5D.py - C:/Users/yashraj/Desktop/RASHI/5D.py (3.12.9)  
File Edit Format Run Options Window Help  
def palindrome(s):  
    return s == s[::-1]  
  
print(palindrome("racecar"))  
print(palindrome("hello"))  
|  
  
IDLE Shell 3.12.9  
File Edit Shell Debug Options Window Help  
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
==== RESTART: C:/Users/yashraj/Desktop/RASHI/5D.py =====  
True  
False  
>>> |
```

Conclusion:

The code is executed successfully.

PRACTICAL 6

Lists

A.

Aim:

The following java code assigns random values (0-10) to each element of array "myArray".

```
int[] myArray = new int[8];  
for(int i = 0; i < myArray.length; i++) {  
    myArray[i] = (int)random(11);  
}
```

Convert the above base code into **python** and write a piece of code in **python** that computes, and display the following on screen.

i. Print the values of each element in myArray ii.

The total of all items in myArray

iii. The number of items in myArray that are greater than 5

iv. The maximum value in array myArray

Source Code:

```
import random
```

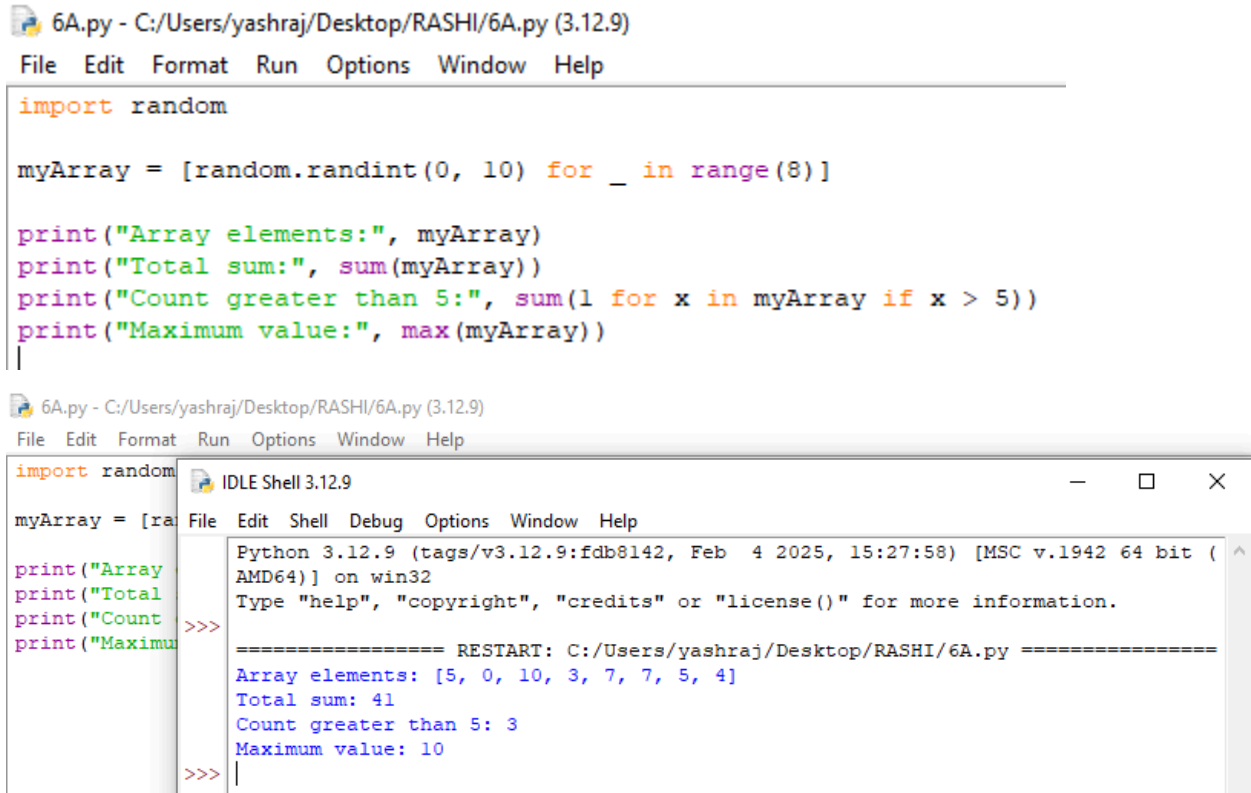
```
myArray = [random.randint(0, 10) for _ in range(8)]
```

```
print("Array elements:", myArray)
```

```
print("Total sum:", sum(myArray))
```

```
print("Count greater than 5:", sum(1 for x in myArray if x > 5))  
print("Maximum value:", max(myArray))
```

Output:



The screenshot displays the Python IDLE 3.12.9 environment. The editor window shows the following code:

```
import random  
  
myArray = [random.randint(0, 10) for _ in range(8)]  
  
print("Array elements:", myArray)  
print("Total sum:", sum(myArray))  
print("Count greater than 5:", sum(1 for x in myArray if x > 5))  
print("Maximum value:", max(myArray))
```

The IDLE Shell window shows the output of the script:

```
Python 3.12.9 (tags/v3.12.9:fd8b142, Feb  4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
  
===== RESTART: C:/Users/yashraj/Desktop/RASHI/6A.py =====  
Array elements: [5, 0, 10, 3, 7, 7, 5, 4]  
Total sum: 41  
Count greater than 5: 3  
Maximum value: 10  
>>>
```

Conclusion:

The code is executed successfully.

B.

Aim:

The following java program declares an int array with 10 elements, each initialized to a random value between (0-30).

```
int[] arr = new int[10];  
for (int i=0; i< arr.length; i++) {  
    arr[i] = (int)random(31);  
    print(arr[i]+" ");  
}
```

Convert the above base code in **Python** and write a program to sort elements of this array in ascending order.

Source Code:

```
import random
```

```
arr = [random.randint(0, 30) for _ in range(10)]
```

```
print("Unsorted array:", arr)
```

```
arr.sort()
```

```
print("Sorted array:", arr)
```


Output:

```
6B.py - C:/Users/yashraj/Desktop/RASHI/6B.py (3.12.9)
File Edit Format Run Options Window Help

import random

arr = [random.randint(0, 30) for _ in range(10)]
print("Unsorted array:", arr)

arr.sort()
print("Sorted array:", arr)

6B.py - C:/Users/yashraj/Desktop/RASHI/6B.py (3.12.9)
File Edit Format Run Options Window Help

import random

arr = [random.randint(0, 30) for _ in range(10)]
print("Unsorted array:", arr)

arr.sort()
print("Sorted array:", arr)

IDLE Shell 3.12.9
File Edit Shell Debug Options Window Help
Python 3.12.9 (tags/v3.12.9:fd8b8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/6B.py =====
Unsorted array: [5, 3, 8, 5, 15, 6, 8, 16, 14, 20]
Sorted array: [3, 5, 5, 6, 8, 8, 14, 15, 16, 20]
>>>
```

Conclusion:

The code is executed successfully.

C.

Aim:

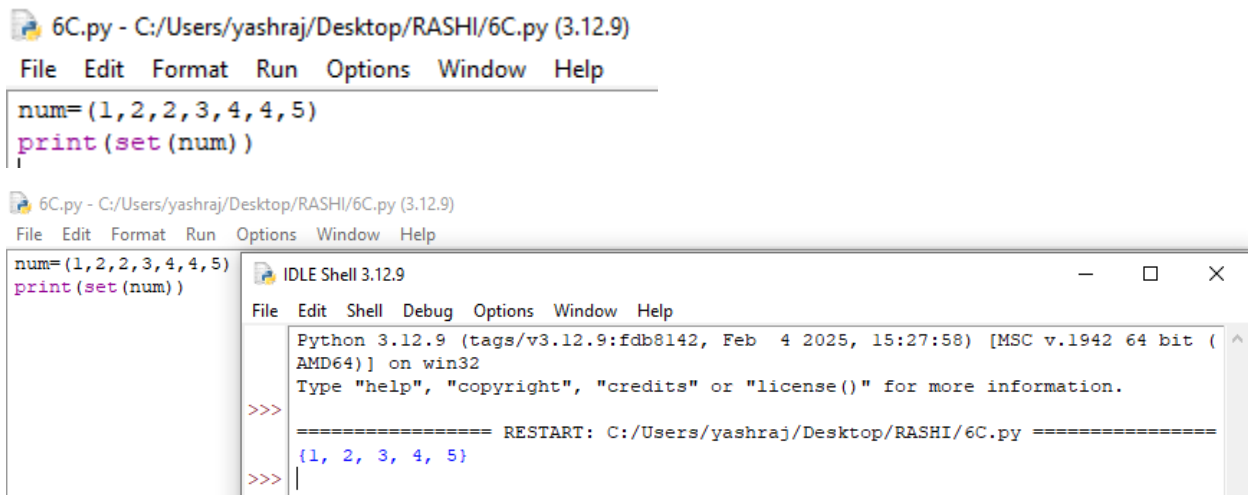
Write a program to remove duplicate items from a list.

Source Code:

```
num=(1,2,2,3,4,4,5)
```

```
print(set(num))
```

Output:



The screenshot displays the Python IDLE 3.12.9 environment. The top window, titled '6C.py - C:/Users/yashraj/Desktop/RASHI/6C.py (3.12.9)', contains the source code: `num=(1,2,2,3,4,4,5)` and `print(set(num))`. The bottom window, titled 'IDLE Shell 3.12.9', shows the execution output. It starts with the Python version and system information, followed by a restart message: `RESTART: C:/Users/yashraj/Desktop/RASHI/6C.py`. The final output is the set `{1, 2, 3, 4, 5}`.

```
6C.py - C:/Users/yashraj/Desktop/RASHI/6C.py (3.12.9)
File Edit Format Run Options Window Help
num=(1,2,2,3,4,4,5)
print(set(num))

IDLE Shell 3.12.9
File Edit Shell Debug Options Window Help
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/yashraj/Desktop/RASHI/6C.py =====
{1, 2, 3, 4, 5}
>>>
```

Conclusion:

The code is executed successfully.

D.

Aim:

Write a program to find common items in 2 lists.

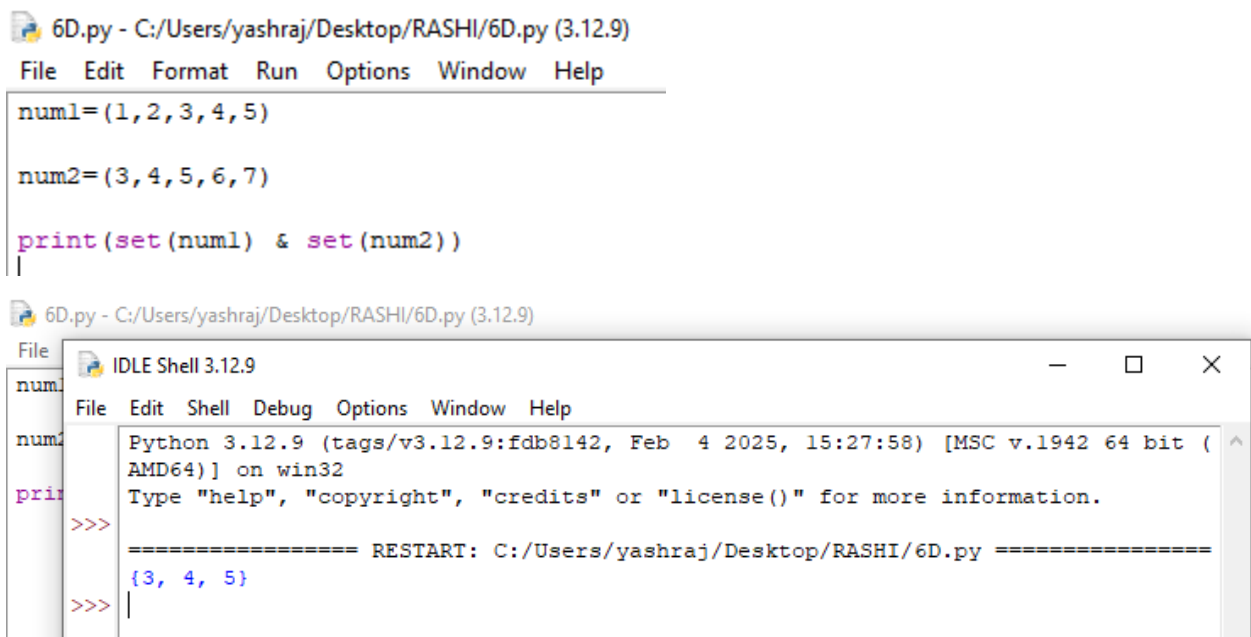
Source Code:

```
num1=(1,2,3,4,5)
```

```
num2=(3,4,5,6,7)
```

```
print(set(num1) & set(num2))
```

Output:



The screenshot displays two windows from the Python IDLE environment. The top window, titled '6D.py - C:/Users/yashraj/Desktop/RASHI/6D.py (3.12.9)', contains the following Python code:

```
File Edit Format Run Options Window Help
num1=(1,2,3,4,5)
num2=(3,4,5,6,7)
print(set(num1) & set(num2))
```

The bottom window, titled 'IDLE Shell 3.12.9', shows the execution output. It includes the standard Python startup message and a restart notice for the file '6D.py'. The output of the print statement is shown as:

```
{3, 4, 5}
```

Conclusion:

The code is executed successfully.

PRACTICAL 7

Tuples and Dictionaries

A.

Aim:

Create and perform the following -

1. An empty tuple
2. Tuple with integers
3. Tuple with mixed data types
4. Nested tuple.
5. Using a while loop print the middle element of the tuple created in 3.

Source Code:

```
empty_tuple=()
print(empty_tuple)
int_tuple=(1,2,3)
print(int_tuple)
mixed_tuple=(1,"Hello",3.14,True)
print(mixed_tuple)
nested_tuple=((1,2,3),("A","B","C"))
print(nested_tuple)
index=len(mixed_tuple)//2
print("Middle Element:",mixed_tuple[index])
```

Output:

```
7A.py - C:/Users/yashraj/Desktop/RASHI/7A.py (3.12.9)
File Edit Format Run Options Window Help
empty_tuple=()

print(empty_tuple)

int_tuple=(1,2,3)

print(int_tuple)

mixed_tuple=(1,"Hello",3.14,True)

print(mixed_tuple)

nested_tuple=((1,2,3),("A","B","C"))

print(nested_tuple)

index=len(mixed_tuple)//2

print("Middle Element:",mixed_tuple[index])
|
```

```
7A.py - C:/Users/yashraj/Desktop/RASHI/7A.py (3.12.9)
File Edit Format Run Options Window Help

empty_tuple=()
print(empty_tuple)

int_tuple=(1,2,3)
print(int_tuple)

mixed_tuple=(1,"Hello",3.14,True)
print(mixed_tuple)

nested_tuple=((1,2,3),("A","B","C"))
print(nested_tuple)

index=len(mixed_tuple)//2
print("Middle Element:",mixed_tuple[index])

IDLE Shell 3.12.9
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/7A.py =====
()
(1, 2, 3)
(1, 'Hello', 3.14, True)
((1, 2, 3), ('A', 'B', 'C'))
Middle Element: 3.14
>>> |
```

Conclusion:

The code is executed successfully.

B.**Aim:**

Join 2 tuples after creation. Find the first and last element of the joined tuple. Swap the elements. Perform slicing between 2nd to 2nd last elements.

Source Code:

```
tuple1=(1,2,3)
tuple2=("a","b","c")
joined_tuple=tuple1+tuple2
print("Joined Tuple:",joined_tuple)
first_element=joined_tuple[0]
last_element=joined_tuple[-1]
print(first_element)
print(last_element)
swapped_tuple=(last_element,)+joined_tuple[1:-1]+(first_element,
)
print("Swapped Tuple:",swapped_tuple)
sliced_tuple=joined_tuple[1:-1]
print("Sliced Tuple:",sliced_tuple)
```

Output:

7B.py - C:/Users/yashraj/Desktop/RASHI/7B.py (3.12.9)

File Edit Format Run Options Window Help

```
tuple1=(1,2,3)

tuple2=("a","b","c")

joined_tuple=tuple1+tuple2

print("Joined Tuple:",joined_tuple)

first_element=joined_tuple[0]

last_element=joined_tuple[-1]

print(first_element)

print(last_element)

swapped_tuple=(last_element,)+joined_tuple[1:-1]+(first_element,)

print("Swapped Tuple:",swapped_tuple)

sliced_tuple=joined_tuple[1:-1]

print("Sliced Tuple:",sliced_tuple)
```

7B.py - C:/Users/yashraj/Desktop/RASHI/7B.py (3.12.9)

File Edit Format Run Options Window Help

tuple1=(1,2,3)

tuple2=("a","b",

joined_tuple=tupl

print("Joined Tup

first_element=jo

last_element=jo

print(first_eleme

print(last_eleme

swapped_tuple=(l

print("Swapped T

sliced_tuple=jo

print("Sliced Tup

IDLE Shell 3.12.9

File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/7B.py =====

Joined Tuple: (1, 2, 3, 'a', 'b', 'c')

1

c

Swapped Tuple: ('c', 2, 3, 'a', 'b', 1)

Sliced Tuple: (2, 3, 'a', 'b')

>>>

>>>

Conclusion:

The code is executed successfully.

C.

Aim:

Create a dictionary named *dict*-

1. Using a for loop print all the key values one by one.
2. Using a while loop print all *values* in dict.
3. Make 3 more dictionaries inside dict named as dict1, dict2, dict3; add values to it.
4. Remove the last inserted key value pair.
5. Copy the dict to a dictionary named ditto.

Source Code:

```
dict_={"Name":"Rashi","Age":19,"Gender":"Female"}
print("Keys in dictionary:")
for key in dict_:
    print(key)
print("Values in dictionary:")
values=list(dict_.values())
i=0
while i<len(values):
    print(values[i])
    i+=1
dict_["dict1"]={"a":1,"b":2}
dict_["dict2"]={"x":10,"y":20}
dict_["dict3"]={"p":"Hello","q":"World"}
```

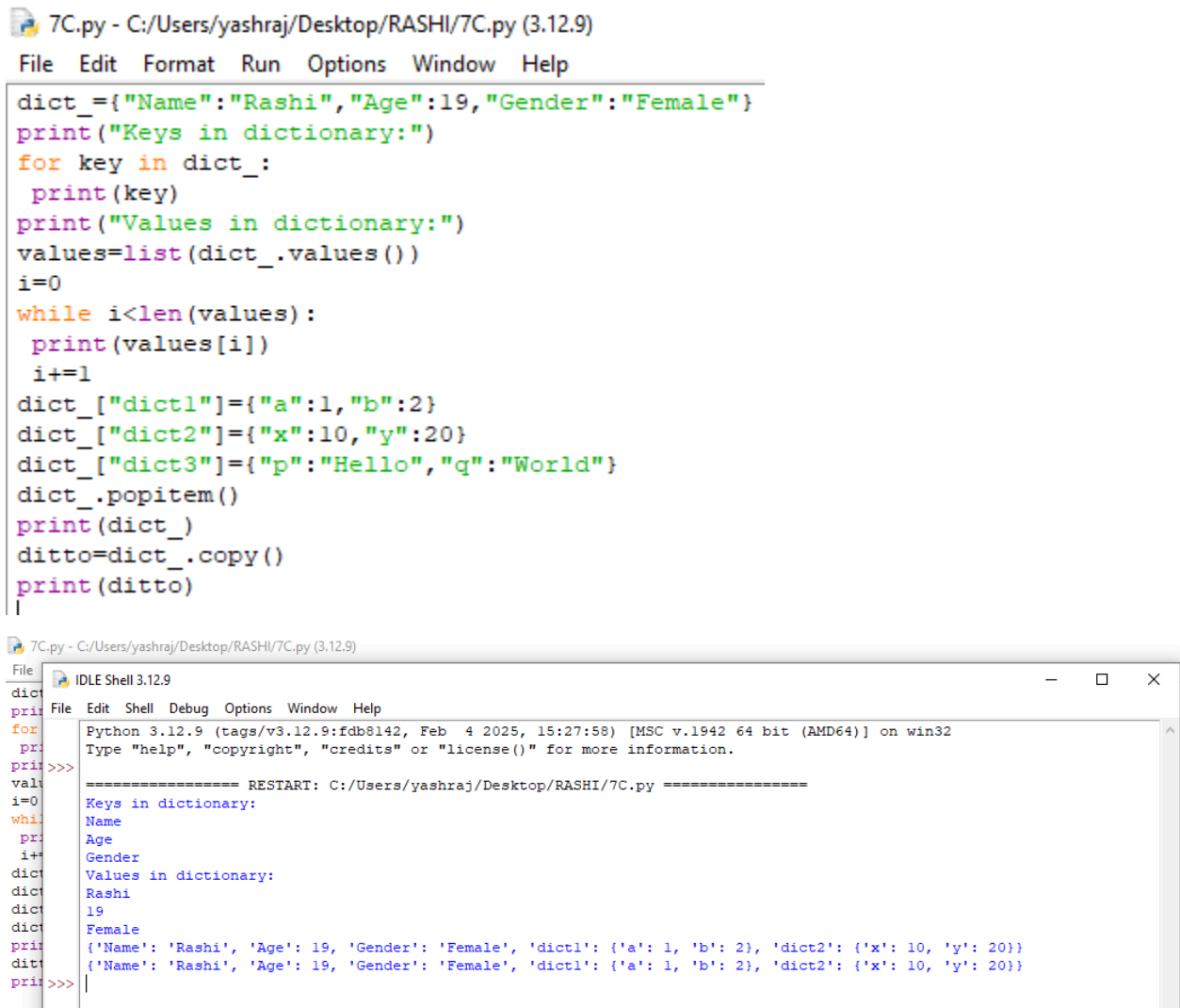

dict_.popitem()

print(dict_)

ditto=dict_.copy()

print(ditto)

Output:



The screenshot displays a Python IDE window titled "7C.py - C:/Users/yashraj/Desktop/RASHI/7C.py (3.12.9)". The code in the editor defines a dictionary 'dict_', iterates over its keys and values, adds three more dictionaries, removes an item, and creates a copy. The output window shows the execution results, including the dictionary's keys, values, and the final state after modifications.

```
dict_={"Name":"Rashi","Age":19,"Gender":"Female"}
print("Keys in dictionary:")
for key in dict_:
    print(key)
print("Values in dictionary:")
values=list(dict_.values())
i=0
while i<len(values):
    print(values[i])
    i+=1
dict_["dict1"]={"a":1,"b":2}
dict_["dict2"]={"x":10,"y":20}
dict_["dict3"]={"p":"Hello","q":"World"}
dict_.popitem()
print(dict_)
ditto=dict_.copy()
print(ditto)
```

Output:

```
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/7C.py =====
Keys in dictionary:
Name
Age
Gender
Values in dictionary:
Rashi
19
Female
{'Name': 'Rashi', 'Age': 19, 'Gender': 'Female', 'dict1': {'a': 1, 'b': 2}, 'dict2': {'x': 10, 'y': 20}}
{'Name': 'Rashi', 'Age': 19, 'Gender': 'Female', 'dict1': {'a': 1, 'b': 2}, 'dict2': {'x': 10, 'y': 20}}
```

Conclusion:

The code is executed successfully.

D.

Aim:

Create a dictionary named Batch having the names as keys and Number of practical attended as value, of the students of this Batch of SYIT. Make a list of defaulters named *default*; for the students who have attended 3 or less practicals. Total number of practicals conducted are 6.

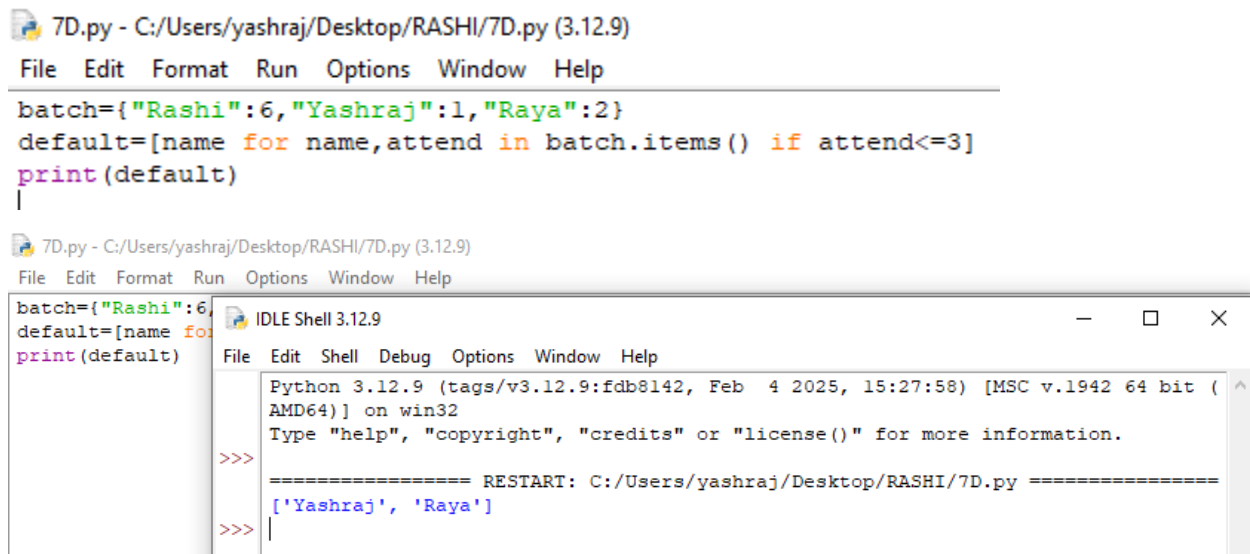
Source Code:

```
batch={"Rashi":6,"Yashraj":1,"Raya":2}
```

```
default=[name for name,attend in batch.items() if attend<=3]
```

```
print(default)
```

Output:



```
7D.py - C:/Users/yashraj/Desktop/RASHI/7D.py (3.12.9)
File Edit Format Run Options Window Help
batch={"Rashi":6,"Yashraj":1,"Raya":2}
default=[name for name,attend in batch.items() if attend<=3]
print(default)
|

7D.py - C:/Users/yashraj/Desktop/RASHI/7D.py (3.12.9)
File Edit Format Run Options Window Help
batch={"Rashi":6,
default=[name for
print(default)

IDLE Shell 3.12.9
File Edit Shell Debug Options Window Help
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/yashraj/Desktop/RASHI/7D.py =====
['Yashraj', 'Raya']
>>>
```

Conclusion:

The code is executed successfully.

PRACTICAL 8

File Handling

A.

Aim:

Create a file named 'py.txt' and write the various file handling methods in it such as- x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+ ; as content to this file. Now perform all these operations on the file py.txt and print the various outputs.

Source Code:

```
with open("py.txt", "w") as f:
```

```
    f.write("x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+")
```

```
with open("py.txt", "a") as f:
```

```
    f.write("\nAppending some text.")
```

```
with open("py.txt", "r") as f:
```

```
    print("\nFile content (R mode)")
```

```
    print(f.read())
```

```
with open("py.txt", "r+") as f:
```

```
    f.write("\nAdded using R+ mode")
```

```
    f.seek(0)
```

```
    print("\nFile content (R+ mode)")
```

```
    print(f.read())
```

```
with open("py.txt", "a+") as f:
```

```
f.write("\nAppending with A+ mode")  
f.seek(0)  
print("\nFile content (A+ mode)")  
print(f.read())
```

Output:

8A.py - C:/Users/yashraj/Desktop/RASHI/8A.py (3.12.9)

File Edit Format Run Options Window Help

```
with open("py.txt", "w") as f:
    f.write("x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+")
with open("py.txt", "a") as f:
    f.write("\nAppending some text.")
with open("py.txt", "r") as f:
    print("\nFile content (R mode)")
    print(f.read())
with open("py.txt", "r+") as f:
    f.write("\nAdded using R+ mode")
    f.seek(0)
    print("\nFile content (R+ mode)")
    print(f.read())
with open("py.txt", "a+") as f:
    f.write("\nAppending with A+ mode")
    f.seek(0)
    print("\nFile content (A+ mode)")
    print(f.read())
```

8A.py - C:/Users/yashraj/Desktop/RASHI/8A.py (3.12.9)

File Edit Format Run Options Window Help

```
with open("py.txt",
f.write("x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+
with open("py.txt",
f.write("\nAppend
with open("py.txt",
print("\nFile co
print(f.read())
with open("py.txt",
f.write("\nAdded
f.seek(0)
print("\nFile co
print(f.read())
with open("py.txt",
f.write("\nAppen
f.seek(0)
print("\nFile co
print(f.read())
```

IDLE Shell 3.12.9

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>> ===== RESTART: C:/Users/yashraj/Desktop/RASHI/8A.py =====

File content (R mode)
x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+
Appending some text.

File content (R+ mode)
Added using R+ mode+, rb, rb+, wb, wb+, ab, ab+
Appending some text.

File content (A+ mode)
Added using R+ mode+, rb, rb+, wb, wb+, ab, ab+
Appending some text.
Appending with A+ mode

>>> |

8A.py - C:/Users/yashraj/Desktop/RASHI/8A.py (3.12.9)

File Edit Format Run Options Window Help

```
with open("py.txt",
f.write("x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+
with open("py.txt",
f.write("\nAppend
with open("py.txt",
print("\nFile co
print(f.read())
with open("py.txt",
f.write("\nAppen
f.seek(0)
print("\nFile co
print(f.read())
with open("py.txt",
f.write("\nAppen
f.seek(0)
print("\nFile co
print(f.read())
```

IDLE Shell 3.12.9

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32

py - Notepad

Added using R+ mode+, rb, rb+, wb, wb+, ab, ab+
Appending some text.
Appending with A+ mode

Conclusion:

The code is executed successfully.

B.

Aim:

Write on the file above using *with()* function. Use a for loop to print each line in py.txt. Now split the content of the file using *split()* function and print the output.

Source Code:

```
with open("py.txt","w") as f:
    f.write("x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+ \n This
is a Second line")
print("Reading each line:")
with open("py.txt", "r") as f:
    for line in f:
        print(line.strip())
with open("py.txt", "r") as f:
    content=f.read()
    words=content.split()
    print("\nSplitting File Content")
    print(words)
```

Output:

```
8B.py - C:/Users/yashraj/Desktop/RASHI/8B.py (3.12.9)
File Edit Format Run Options Window Help

with open("py.txt","w") as f:
    f.write("x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+ \n This is a Second line")
print("Reading each line:")
with open("py.txt", "r") as f:
    for line in f:
        print(line.strip())
with open("py.txt", "r") as f:
    content=f.read()
    words=content.split()
    print("\nSplitting File Content")
    print(words)
```

```
8B.py - C:/Users/yashraj/Desktop/RASHI/8B.py (3.12.9)
File Edit Format Run Options Window Help

with op
f.w
print("
with op
for
with op
cor
wor
pri
pri

IDLE Shell 3.12.9
File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>>
===== RESTART: C:/Users/yashraj/Desktop/RASHI/8B.py =====
Reading each line:
x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+
This is a Second line

Splitting File Content
['x,', 'a,', 'r,', 'w,', 'r+', 'w+', 'a+', 'rb,', 'rb+', 'wb,', 'wb+', 'ab,', 'ab+', 'This', 'is', 'a', 'Second', 'line']
>>>
```

```
8B.py - C:/Users/yashraj/Desktop/RASHI/8B.py (3.12.9)
File Edit Format Run Options Window Help

with open("py.txt",
f.write("x, a,
print("Reading each
with open("py.txt",
for line in f:
    print(line.s
with open("py.txt",
content=f.read()
words=content.sp
print("\nSplitting
print(words)

py - Notepad
File Edit Format View Help
x, a, r, w, r+, w+, a+, rb, rb+, wb, wb+, ab, ab+
This is a Second line
```

Conclusion:

The code is executed successfully.

PRACTICAL 9

Classes and Objects

A.

Aim:

Create a Class named Person, use the `__init__()` function to assign values for name, age and gender Create an object and print the values.

Source Code:

```
class Person:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender
    def display_info(self):
        print("My name is", self.name)
        print("My age is", self.age)
        print("My gender is", self.gender)
obj = Person("Rashi", 19, "Female")
obj.display_info()
```

Output:

9A.py - C:/Users/yashraj/Desktop/RASHI/9A.py (3.12.9)

File Edit Format Run Options Window Help

```
class Person:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender
    def display_info(self):
        print("My name is", self.name)
        print("My age is", self.age)
        print("My gender is", self.gender)
obj = Person("Rashi", 19, "Female")
obj.display_info()
```

9A.py - C:/Users/yashraj/Desktop/RASHI/9A.py (3.12.9)

File Edit Format Run Options Window Help

class Pers
def

se File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license()" for more information.

>>>

===== RESTART: C:/Users/yashraj/Desktop/RASHI/9A.py =====

My name is Rashi

My age is 19

My gender is Female

>>>

Conclusion:

The code is executed successfully.

B.

Aim:

Create a Child Class Student of the above Class Person which will inherit it's properties and methods. Use `__init__()` function and add a few properties and methods such as `fname` and `lname` to Student Class. Check if the Child Class function overrides the function of Parent Class Person. If this happens then add a call to the parent's `__init__()` function OR use the `super()` function for the same task.

Source Code:

```
class Person:

    def __init__(self, name, age, gender):

        self.name = name

        self.age = age

        self.gender = gender

    def display_info(self):

        print("My name is", self.name)

        print("My age is", self.age)

        print("My gender is", self.gender)

class Student(Person):

    def __init__(self, name, age, gender, fname, lname):

        super().__init__(name, age, gender)
```

```
self.fname = fname
```

```
self.lname = lname
```

```
def display_student_info(self):
```

```
    self.display_info()
```

```
    print("My First name is", self.fname)
```

```
    print("My Last name is", self.lname)
```

```
obj = Student("Rashi", 19, "Female", "Rashi", "Sawardekar")
```

```
obj.display_student_info()
```

Output:

9B.py - C:/Users/yashraj/Desktop/RASHI/9B.py (3.12.9)

File Edit Format Run Options Window Help

```
class Person:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender

    def display_info(self):
        print("My name is", self.name)
        print("My age is", self.age)
        print("My gender is", self.gender)

class Student(Person):
    def __init__(self, name, age, gender, fname, lname):
        super().__init__(name, age, gender)
        self.fname = fname
        self.lname = lname

    def display_student_info(self):
        self.display_info()
        print("My First name is", self.fname)
        print("My Last name is", self.lname)

obj = Student("Rashi", 19, "Female", "Rashi", "Sawardekar")

obj.display_student_info()
```

9B.py - C:/Users/yashraj/Desktop/RASHI/9B.py (3.12.9)

File Edit Format Run Options Window Help

```
class Person:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender

    def display_info(self):
        print("My name is", self.name)
        print("My age is", self.age)
        print("My gender is", self.gender)

class Student(Person):
    def __init__(self, name, age, gender, fname, lname):
        super().__init__(name, age, gender)
        self.fname = fname
        self.lname = lname

    def display_student_info(self):
        self.display_info()
        print("My First name is", self.fname)
        print("My Last name is", self.lname)

obj = Student("Rashi", 19, "Female", "Rashi", "Sawardekar")

obj.display_student_info()
```

IDLE Shell 3.12.9

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/9B.py =====

>>> My name is Rashi
My age is 19
My gender is Female
My First name is Rashi
My Last name is Sawardekar
>>>

Conclusion:

The code is executed successfully.

PRACTICAL 10

Regular Expressions

A.

Aim:

Add a property `graduationyear` to the student class using `self` and pass a value 2023 to it. Add a method called *welcome()* to the student class, which prints ("Welcome", `fname`, `lname`, "to the class of" , `graduationyear`)

Source Code:

```
class Student:

    def __init__(self, fname, lname, graduation_year=2023):

        self.fname = fname

        self.lname = lname

        self.graduation_year = graduation_year

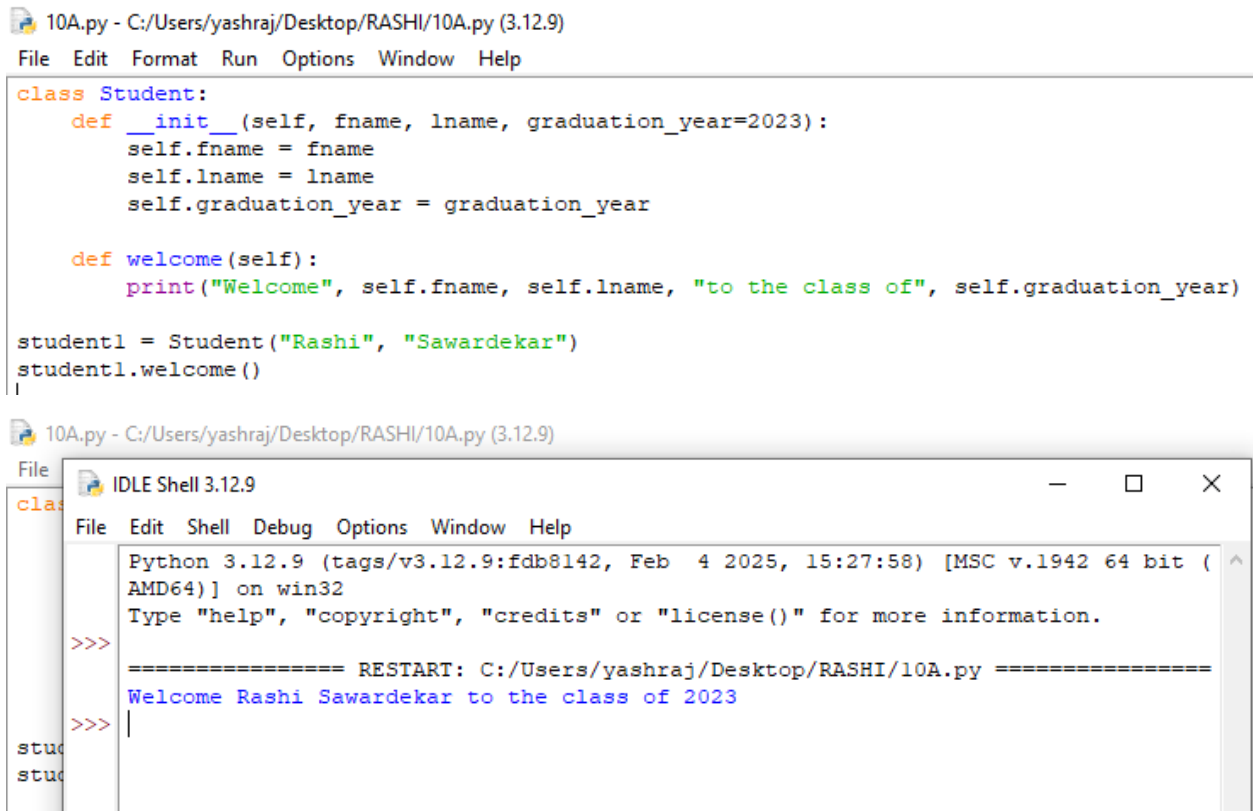

    def welcome(self):

        print("Welcome", self.fname, self.lname, "to the class of",
self.graduation_year)


student1 = Student("Rashi", "Sawardekar")

student1.welcome()
```

Output:



The image shows two windows from a Python IDE. The top window, titled '10A.py - C:/Users/yashraj/Desktop/RASHI/10A.py (3.12.9)', contains the following Python code:

```
class Student:
    def __init__(self, fname, lname, graduation_year=2023):
        self.fname = fname
        self.lname = lname
        self.graduation_year = graduation_year

    def welcome(self):
        print("Welcome", self.fname, self.lname, "to the class of", self.graduation_year)

student1 = Student("Rashi", "Sawardekar")
student1.welcome()
```

The bottom window, titled 'IDLE Shell 3.12.9', shows the execution output:

```
Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/yashraj/Desktop/RASHI/10A.py =====
Welcome Rashi Sawardekar to the class of 2023
>>>
```

Conclusion:

The code is executed successfully.

B.

Aim:

Import re, Use RegEx methods findall(), search() or match() to check and print if the *fname* or *lname* has 'a' in it, also check if graduationyear has '0' in it.

Source Code:

```
import re

class Student:

    def __init__(self, fname, lname, graduation_year=2023):

        self.fname = fname

        self.lname = lname

        self.graduation_year = str(graduation_year)

    def check_name(self):

        if re.search(r"a",self.fname,re.IGNORECASE) or
re.search(r"a",self.lname,re.IGNORECASE):

            print("The name contains the letter 'a'")

        else:

            print("The name does not contain the letter 'a'")

    def check_num(self):

        if re.findall(r"0",self.graduation_year):

            print("The graduation year contains the digit '0'")

        else:
```

```
        print("The graduation year does not contain the digit  
'0'")
```

```
    def welcome(self):
```

```
        print("Welcome", self.fname, self.lname, "to the class of",  
self.graduation_year)
```

```
student1=Student("Rashi","Sawardekar")
```

```
student1.welcome()
```

```
student1.check_name()
```

```
student1.check_num()
```

Output:

10B.py - C:/Users/yashraj/Desktop/RASHI/10B.py (3.12.9)

File Edit Format Run Options Window Help

```
import re
class Student:
    def __init__(self, fname, lname, graduation_year=2023):
        self.fname = fname
        self.lname = lname
        self.graduation_year = str(graduation_year)
    def check_name(self):
        if re.search(r"a",self.fname,re.IGNORECASE) or re.search(r"a",self.lname,re.IGNORECASE):
            print("The name contains the letter 'a'")
        else:
            print("The name does not contain the letter 'a'")
    def check_num(self):
        if re.findall(r"0",self.graduation_year):
            print("The graduation year contains the digit '0'")
        else:
            print("The graduation year does not contain the digit '0'")
    def welcome(self):
        print("Welcome", self.fname, self.lname, "to the class of", self.graduation_year)

student1=Student("Rashi","Sawardekar")
student1.welcome()
student1.check_name()
student1.check_num()
```

10B.py - C:/Users/yashraj/Desktop/RASHI/10B.py (3.12.9)

File Edit Format Run Options Window Help

```
import re
class Student:
    def __init__(self, fname, lname, graduation_year=2023):
        self.fname = fname
        self.lname = lname
        self.graduation_year = str(graduation_year)
    def check_name(self):
        if re.search(r"a",self.fname,re.IGNORECASE) or re.search(r"a",self.lname,re.IGNORECASE):
            print("The name contains the letter 'a'")
        else:
            print("The name does not contain the letter 'a'")
    def check_num(self):
        if re.findall(r"0",self.graduation_year):
            print("The graduation year contains the digit '0'")
        else:
            print("The graduation year does not contain the digit '0'")
    def welcome(self):
        print("Welcome", self.fname, self.lname, "to the class of", self.graduation_year)

student1=Student("Rashi","Sawardekar")
student1.welcome()
student1.check_name()
student1.check_num()
```

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/10B.py =====
Welcome Rashi Sawardekar to the class of 2023
The name contains the letter 'a'
The graduation year contains the digit '0'
|

Conclusion:

The code is executed successfully.

PRACTICAL 11

GUI using Python

Aim:

Create an admission form to display all the elements of tkinter module, validate the user input (password and username through a textbox)

Source Code:

```
import tkinter as tk

from tkinter import messagebox

def validate_login():

    username = user_entry.get()

    password = pass_entry.get()

    if username == "admin" and password == "1234":

        messagebox.showinfo("Login", "Login successful,
Welcome to the Admission System")

    else:

        messagebox.showerror("Login", "Invalid username or
password")

pl=tk.Tk()
```

```
pl.title("Admission Page")
```

```
pl.geometry("300x200")
```

```
user_label=tk.Label(pl,text="username:")
```

```
user_label.grid(row=0,column=0,pady=10,padx=5)
```

```
user_entry=tk.Entry(pl)
```

```
user_entry.grid(row=0,column=1)
```

```
pass_label=tk.Label(pl,text="Password:")
```

```
pass_label.grid(row=1,column=0,pady=10,padx=5)
```

```
pass_entry=tk.Entry(pl,show="*")
```

```
pass_entry.grid(row=1,column=1)
```

```
btn=tk.Button(pl,text="Login",command=validate_login)
```

```
btn.grid(row=2,column=0,columnspan=2,pady=10)
```

```
pl.mainloop()
```

Output:

11.py - C:/Users/yashraj/Desktop/RASHI/11.py (3.12.9)

File Edit Format Run Options Window Help

```
import tkinter as tk
from tkinter import messagebox

def validate_login():
    username = user_entry.get()
    password = pass_entry.get()
    if username == "admin" and password == "1234":
        messagebox.showinfo("Login", "Login successful, Welcome to the Admission System")
    else:
        messagebox.showerror("Login", "Invalid username or password")

pl=tk.Tk()
pl.title("Admission Page")
pl.geometry("300x200")

user_label=tk.Label(pl,text="username:")
user_label.grid(row=0,column=0,pady=10,padx=5)
user_entry=tk.Entry(pl)
user_entry.grid(row=0,column=1)

pass_label=tk.Label(pl,text="Password:")
pass_label.grid(row=1,column=0,pady=10,padx=5)
pass_entry=tk.Entry(pl,show="*")
pass_entry.grid(row=1,column=1)

btn=tk.Button(pl,text="Login",command=validate_login)
btn.grid(row=2,column=0,columnspan=2,pady=10)

pl.mainloop()
```

9A.py - C:/Users/yashraj/Desktop/RASHI/9A.py (3.12.9)

File

Admission Page

username:

Password:

Login

obj

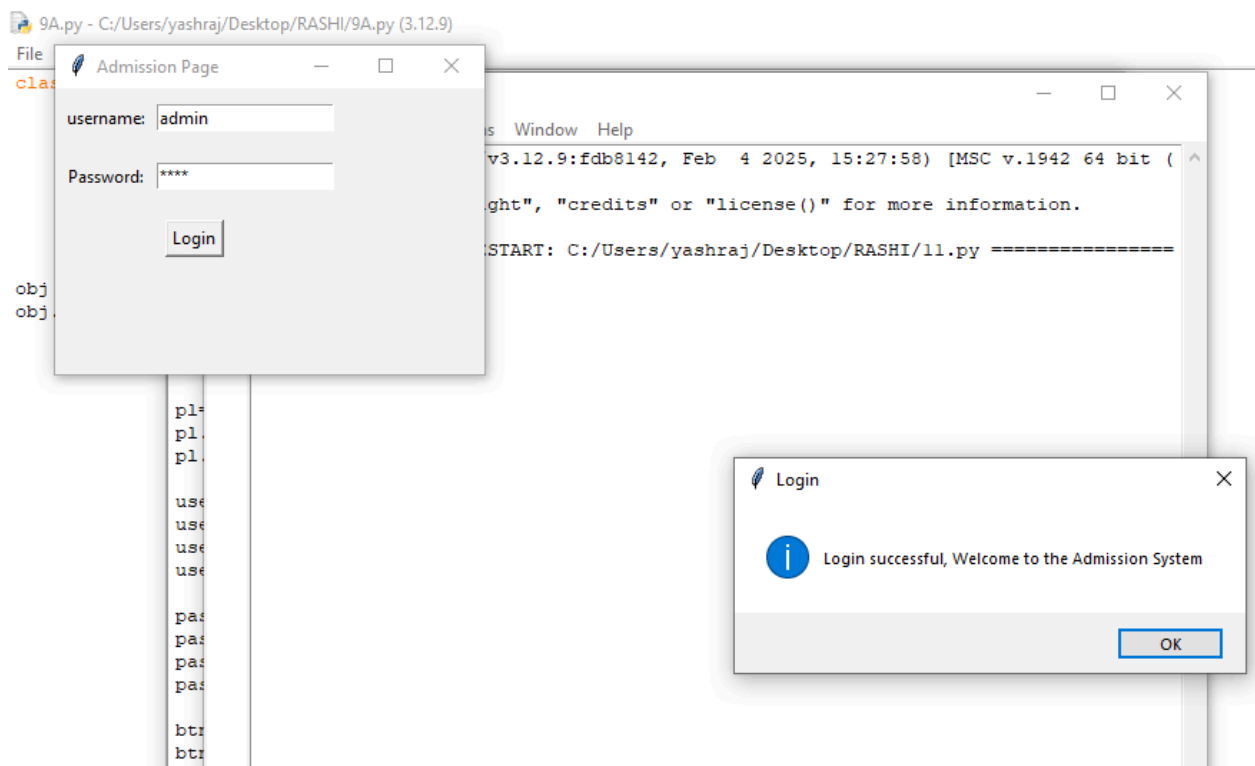
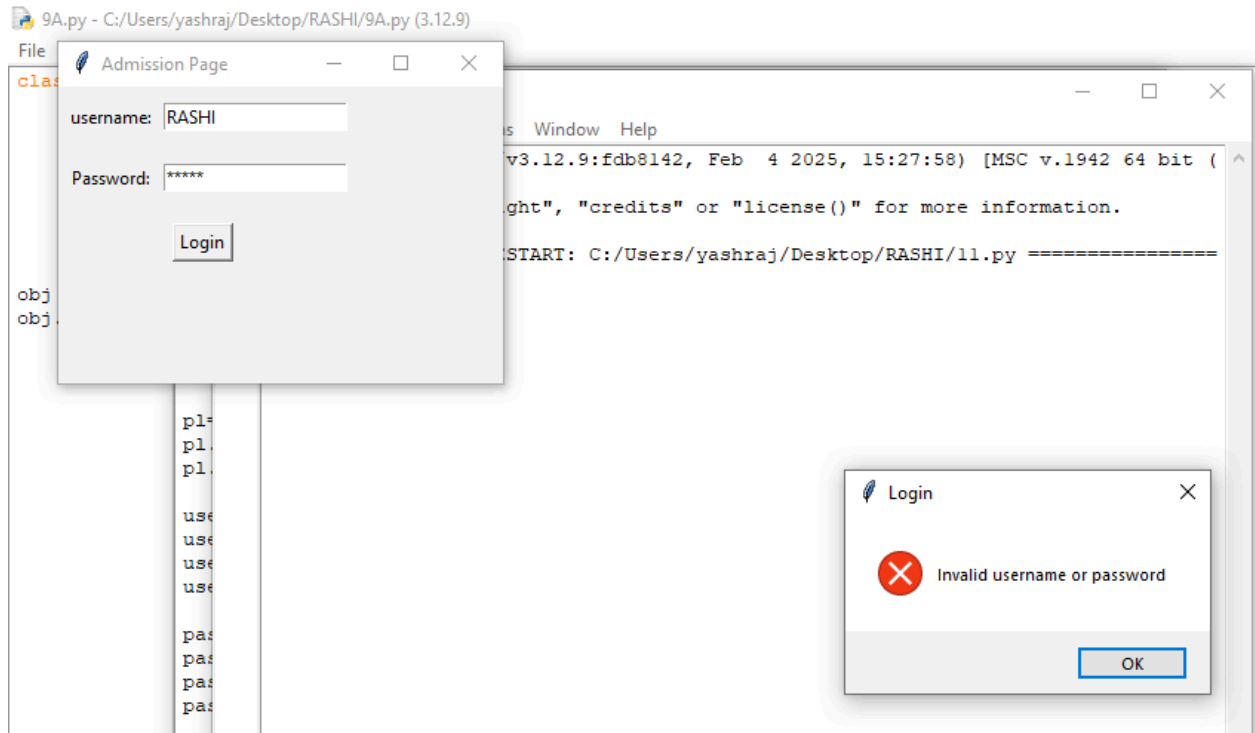
pl=
pl=
pl=

s Window Help

v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)]

ght", "credits" or "license()" for more information.

START: C:/Users/yashraj/Desktop/RASHI/11.py =====



Conclusion:

The code is executed successfully.

PRACTICAL 12

Database Connectivity using Python

Aim:

Using a connector of MySql establish a database connection and using cursor Create, Update, and Delete database entries for Student_id, name, class, year, CGPA, gender.

Source Code:

```
import mysql.connector

conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='yashraj@123',
    database='testdb'
)

cursor = conn.cursor()

cursor.execute("""
    CREATE TABLE IF NOT EXISTS students (
        student_id INT AUTO_INCREMENT PRIMARY KEY,
```



```
        name VARCHAR(250),
        student_class VARCHAR(250),
        year INT,
        CGPA FLOAT,
        gender VARCHAR(25)
    )
    """)
```

```
conn.commit()
```

```
print("Table 'students' is ready.")
```

```
def insert_student(name, student_class, year, cgpa, gender):
```

```
    cursor.execute("INSERT INTO students (name,
student_class, year, cgpa, gender) VALUES (%s, %s, %s, %s,
%s)",
```

```
        (name, student_class, year, cgpa, gender))
```

```
    conn.commit()
```

```
    print(f" Student Record Inserted: {name}")
```

```
def update_student(student_id, new_cgpa):
```

```
    cursor.execute("UPDATE students SET CGPA = %s WHERE
student_id = %s", (new_cgpa, student_id))
```

```
    conn.commit()
```

```
print(f" Student Record Updated! ID: {student_id}, New  
CGPA: {new_cgpa}")
```

```
def delete_student(student_id):  
    cursor.execute("DELETE FROM students WHERE  
student_id = %s", (student_id,))  
    conn.commit()  
    print(f" Student Record Deleted! ID: {student_id}")
```

```
def fetch_students():  
    cursor.execute("SELECT * FROM students")  
    students = cursor.fetchall()  
    print("\n Student Records:")  
    for student in students:  
        print(student)
```

```
insert_student("Rashi", "FYIT", 2025, 9.8, "Female")  
insert_student("Raya", "SYCS", 2024, 9.6, "Female")  
insert_student("Yashraj", "TYCS", 2023, 9.7, "Male")
```

```
update_student(2, 9.85)  
delete_student(1)
```

```
delete_student(3)
```

```
fetch_students()
```

```
cursor.close()
```

```
conn.close()
```

```
print("Database Connection Closed.")
```

Output:

12.py - C:/Users/yashraj/Desktop/RASHI/12.py (3.12.9)

File Edit Format Run Options Window Help

```
import mysql.connector

conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='yashraj@123',
    database='testdb'
)

cursor = conn.cursor()

cursor.execute("""
    CREATE TABLE IF NOT EXISTS students (
        student_id INT AUTO_INCREMENT PRIMARY KEY,
        name VARCHAR(250),
        student_class VARCHAR(250),
        year INT,
        CGPA FLOAT,
        gender VARCHAR(25)
    )
""")
conn.commit()
print("Table 'students' is ready.")

def insert_student(name, student_class, year, cgpa, gender):
    cursor.execute("INSERT INTO students (name, student_class, year, cgpa, gender) VALUES (%s, %s, %s, %s, %s)",
        (name, student_class, year, cgpa, gender))
    conn.commit()
    print(f"Student Record Inserted: {name}")

def update_student(student_id, new_cgpa):
    cursor.execute("UPDATE students SET CGPA = %s WHERE student_id = %s", (new_cgpa, student_id))
    conn.commit()
    print(f"Student Record Updated! ID: {student_id}, New CGPA: {new_cgpa}")

def delete_student(student_id):
    cursor.execute("DELETE FROM students WHERE student_id = %s", (student_id,))
    conn.commit()
    print(f"Student Record Deleted! ID: {student_id}")

def fetch_students():
    .....
```

12.py - C:/Users/yashraj/Desktop/RASHI/12.py (3.12.9)

File Edit Format Run Options Window Help

```
"""
conn.commit()
print("Table 'students' is ready.")

def insert_student(name, student_class, year, cgpa, gender):
    cursor.execute("INSERT INTO students (name, student_class, year, cgpa, gender) VALUES (%s, %s, %s, %s, %s)",
                    (name, student_class, year, cgpa, gender))
    conn.commit()
    print(f"Student Record Inserted: {name}")

def update_student(student_id, new_cgpa):
    cursor.execute("UPDATE students SET CGPA = %s WHERE student_id = %s", (new_cgpa, student_id))
    conn.commit()
    print(f"Student Record Updated! ID: {student_id}, New CGPA: {new_cgpa}")

def delete_student(student_id):
    cursor.execute("DELETE FROM students WHERE student_id = %s", (student_id,))
    conn.commit()
    print(f"Student Record Deleted! ID: {student_id}")

def fetch_students():
    cursor.execute("SELECT * FROM students")
    students = cursor.fetchall()
    print("\n Student Records:")
    for student in students:
        print(student)

insert_student("Rashi", "FYIT", 2025, 9.8, "Female")
insert_student("Raya", "SYCS", 2024, 9.6, "Female")
insert_student("Yashraj", "TYCS", 2023, 9.7, "Male")

update_student(2, 9.85)
delete_student(1)
delete_student(3)

fetch_students()

cursor.close()
conn.close()
print("Database Connection Closed.")
```

12.py - C:/Users/yashraj/Desktop/RASHI/12.py (3.12.9)

File Edit Format Run Options Window Help

import mysql

conn = mysql

host='loc

user='roo

password=

database=

)

cursor = conn

cursor.execut

CREATE TA

stude

name

stude

year

CGPA

gende

)

"""

conn.commit()

IDLE Shell 3.12.9

File Edit Shell Debug Options Window Help

Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

===== RESTART: C:/Users/yashraj/Desktop/RASHI/12.py =====

Table 'students' is ready.

Student Record Inserted: Rashi

Student Record Inserted: Raya

Student Record Inserted: Yashraj

Student Record Updated! ID: 2, New CGPA: 9.85

Student Record Deleted! ID: 1

Student Record Deleted! ID: 3

Student Records:

(2, 'Raya', 'SYCS', 2024, 9.85, 'Female')

Database Connection Closed.

>>>

```
12.py - C:/Users/yashraj/Desktop/RASHI/12.py (3.12.9)
File Edit Format Run Options Window Help

import mysql.connector
conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='root',
    database='testdb'
)
cursor = conn.cursor()
cursor.execute("""
CREATE TABLE students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(250),
    student_class VARCHAR(250),
    year INT,
    CGPA FLOAT,
    gender VARCHAR(25)
)
""")
conn.commit()
print("Table created successfully")

def insert_student():
    cursor.execute("""
INSERT INTO students (name, student_class, year, CGPA, gender)
VALUES ('Raya', 'SYCS', 2024, 9.85, 'Female')
""")
    conn.commit()
    print("Student inserted successfully")

def update_student():
    cursor.execute("""
UPDATE students
SET CGPA = 9.9
WHERE student_id = 2
""")
    conn.commit()
    print("Student updated successfully")

def delete_student():
    cursor.execute("""
DELETE FROM students
WHERE student_id = 2
""")
    conn.commit()
    print("Student deleted successfully")

if __name__ == '__main__':
    insert_student()
    update_student()
    delete_student()
```

Python 3.12.9 (tags/v3.12.9:fd8b142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

MySQL 8.0 Command Line Client
Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
mysql> USE testdb;
Database changed
mysql> DESC students;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| student_id | int | NO | PRI | NULL | auto_increment |
| name | varchar(250) | YES | | NULL | |
| student_class | varchar(250) | YES | | NULL | |
| year | int | YES | | NULL | |
| CGPA | float | YES | | NULL | |
| gender | varchar(25) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql> SELECT * FROM students;
+-----+-----+-----+-----+-----+-----+
| student_id | name | student_class | year | CGPA | gender |
+-----+-----+-----+-----+-----+-----+
| 2 | Raya | SYCS | 2024 | 9.85 | Female |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

MySQL 8.0 Command Line Client

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
mysql> USE testdb;
Database changed
mysql> DESC students;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| student_id | int | NO | PRI | NULL | auto_increment |
| name | varchar(250) | YES | | NULL | |
| student_class | varchar(250) | YES | | NULL | |
| year | int | YES | | NULL | |
| CGPA | float | YES | | NULL | |
| gender | varchar(25) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql> SELECT * FROM students;
+-----+-----+-----+-----+-----+-----+
| student_id | name | student_class | year | CGPA | gender |
+-----+-----+-----+-----+-----+-----+
| 2 | Raya | SYCS | 2024 | 9.85 | Female |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

Conclusion:

The code is executed successfully.

