

# 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)
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\1.py =====
IMPLICIT TYPE Casting
<class 'int'>
<class 'float'>
<class 'str'>
<class 'bool'>
<class 'complex'>
EXPLICIT TYPE CASTING
<class 'str'>
J=int('Y')
Y
<class 'int'>
M=float('21.0')
21.0
print(type(M))
21
Y=str(3)
<class 'str'>
print(ord(Y))
3
print("Ascii Value of Character Y is: ",ord(Y))
Ascii Value of character digit 1 is: 49
g=int('1921',2)
The Value of 1921 in other number system
print(bin(g))
0b1110000001 in binary number
alp=oct(g)
Oo3601 in octal
num=hex(g)
0x781 in hexadecimal
print(type(g))
```

## Conclusion:

The code is executed successfully.

# PRACTICAL 2

## Conditions

### A.

#### Aim:

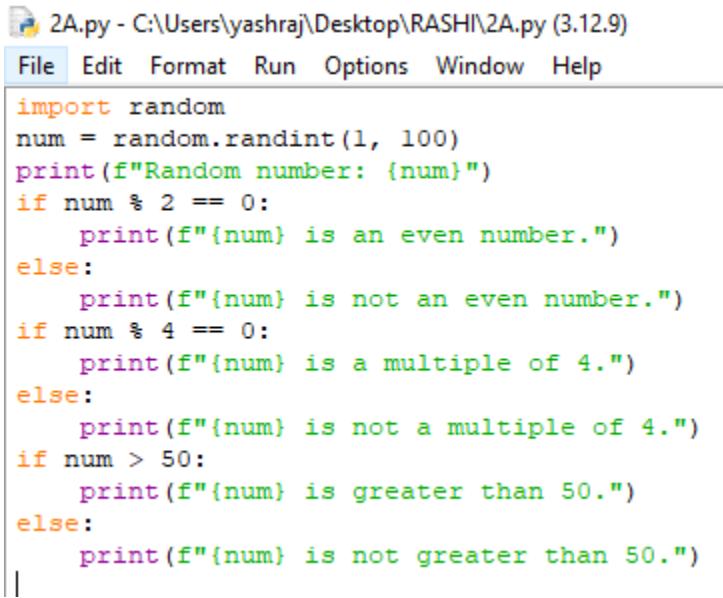
Write a Python program that:

- 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

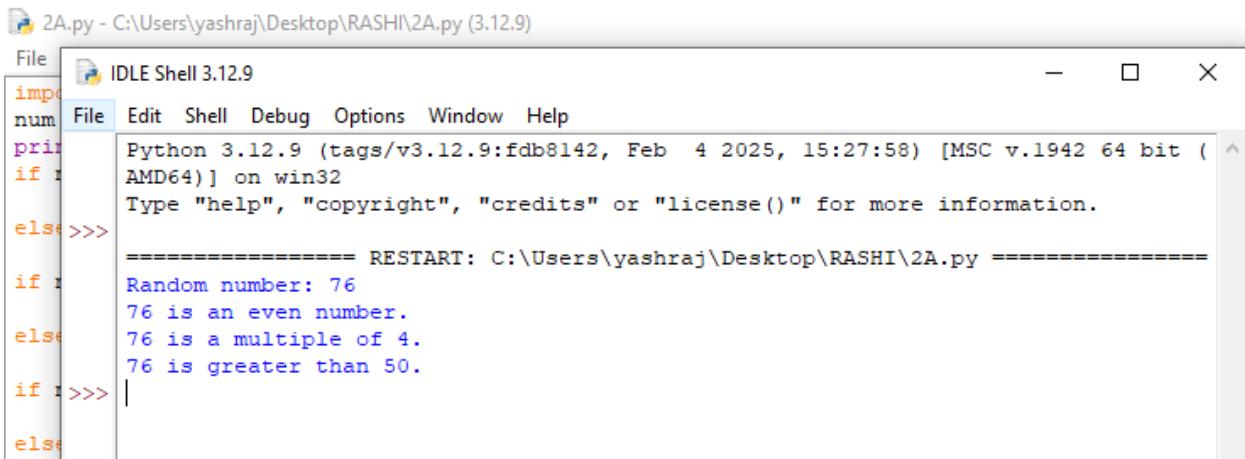
#### Source Code:

```
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 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\2A.py =====
if r
    Random number: 76
    76 is an even number.
else
    76 is a multiple of 4.
    76 is greater than 50.
if r>>> |
```

## 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_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.")

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.

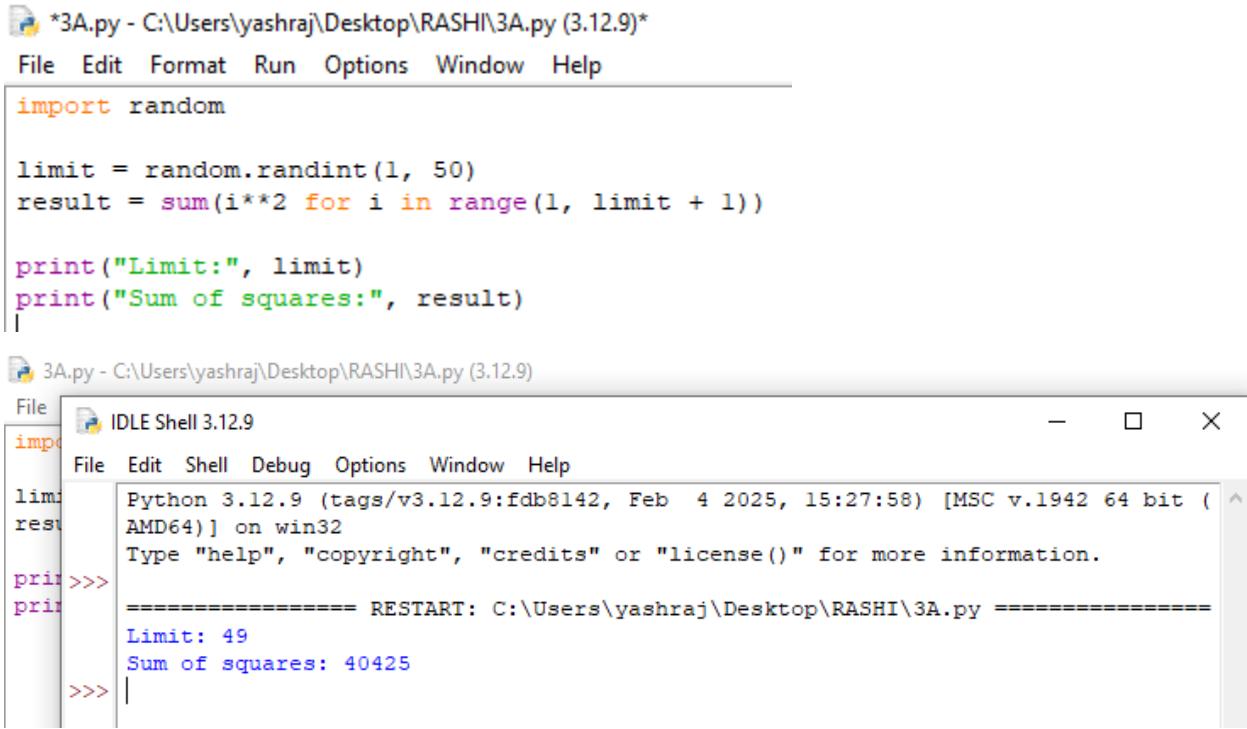
### Aim:

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 shows a Python IDE interface with two windows. The top window is titled '3A.py - C:\Users\yashraj\Desktop\RASHI\3A.py (3.12.9)\*' and 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 is titled '3A.py - C:\Users\yashraj\Desktop\RASHI\3A.py (3.12.9)' and is titled 'IDLE Shell 3.12.9'. It shows the execution of the script:

```
File Edit Shell Debug Options Window Help
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.

>>> print("Limit:", limit)
>>> print("Sum of squares:", result)
=====
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.

$$A = a_0 + \sum (a \cos \cos n\pi x L + b \sin \sin n\pi x L) 30n=5$$

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:

The image shows two windows. The top window is a code editor with the file 3B.py open. The code calculates a value A based on input x, using a series of trigonometric functions. The bottom window is the IDLE Shell 3.12.9, where the script is run. It shows the script's contents on the left and the shell's output on the right, including the calculated result of 1445.9932425806612.

```
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
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\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:

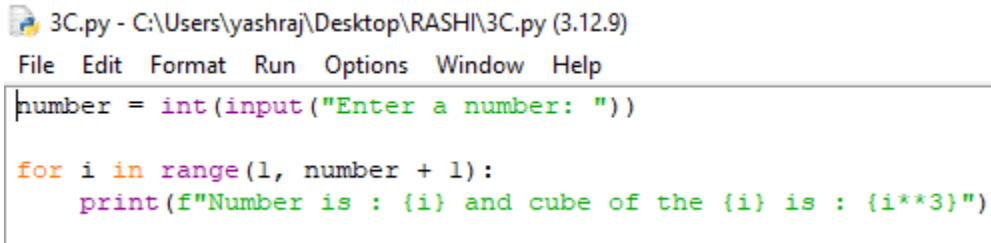
Number is : 1 and cube of the 1 is :1  
Number is : 2 and cube of the 2 is :8  
Number is : 3 and cube of the 3 is :27  
Number is : 4 and cube of the 4 is :64  
Number is : 5 and cube of the 5 is :125

### **Source Code:**

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

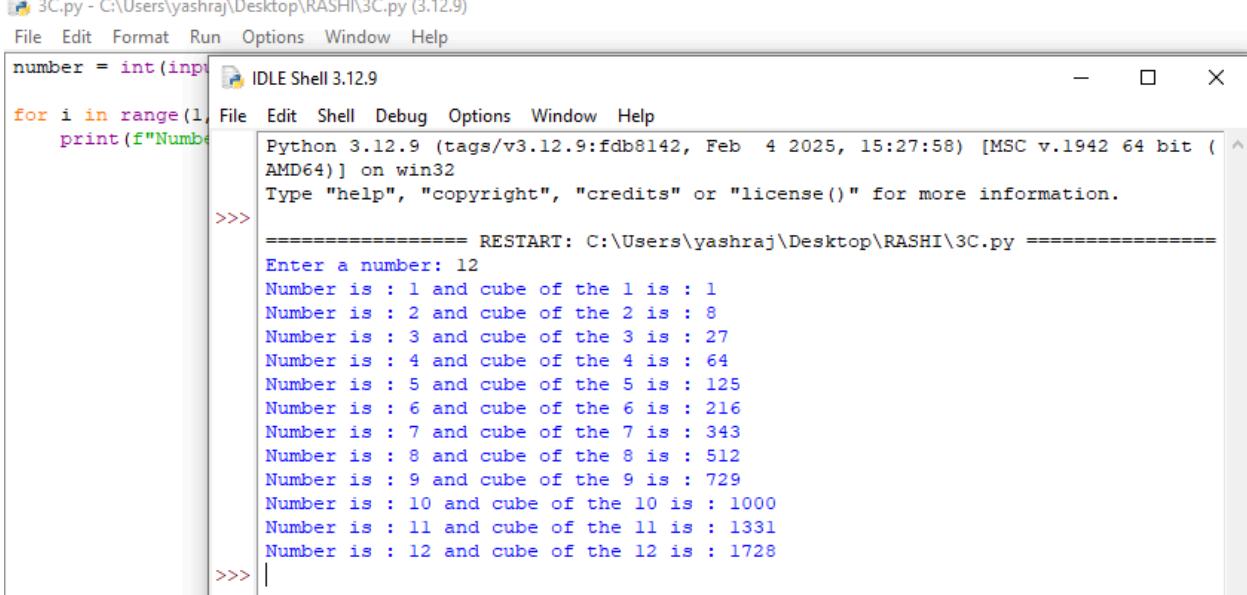
for i in range(1, number + 1):
    print(f"Number is : {i} and cube of the {i} is : {i**3}")
```

## Output:



```
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}")
```

```
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\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):
```

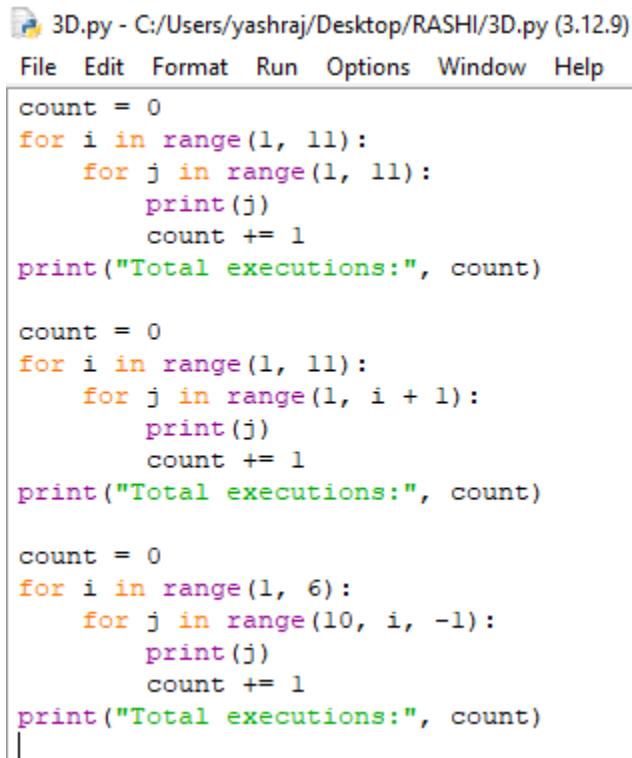
```

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:



The screenshot shows a code editor window with three identical Python scripts named "3D.py" side-by-side. Each script contains the same three pieces of code: a single-line print statement, a loop that increments a counter, and a final print statement outputting the total number of executions. The code is color-coded with syntax highlighting.

```

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)

```

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

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/3D.py =====

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

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
10
1
2
3
4
5
6
7
8
9
10
1
2
3
4
5
6
7
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  
10  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
Total executions: 100  
1  
1  
2  
1  
2  
3  
1  
^

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

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

The screenshot shows a dual-pane Python IDE interface. The left pane displays the source code of a file named '3D.py' located at 'C:/Users/yashraj/Desktop/RASHI/3D.py (3.12.9)'. The right pane shows the execution results in the IDLE Shell 3.12.9 window.

```
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
Total executions: 55
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
```

```
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
9
10
Total executions: 55
10
9
8
7
6
5
4
3
2
10
9
8
7
6
5
4
3
10
9
8
7
6
5
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 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,
    for j in range(i,
        print(i,
    print()
print("Pattern b">>>>
for i in range(1,
    for j in range(
        print(i,
    print()
print("Pattern c")
for i in range(1,
    for j in range(>>>
        print(j,
    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 4 4 4 5 5 5 5 6 6 6 6 6  
Pattern b:  
1 1 1 1 1 2 2 2 2 3 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:**

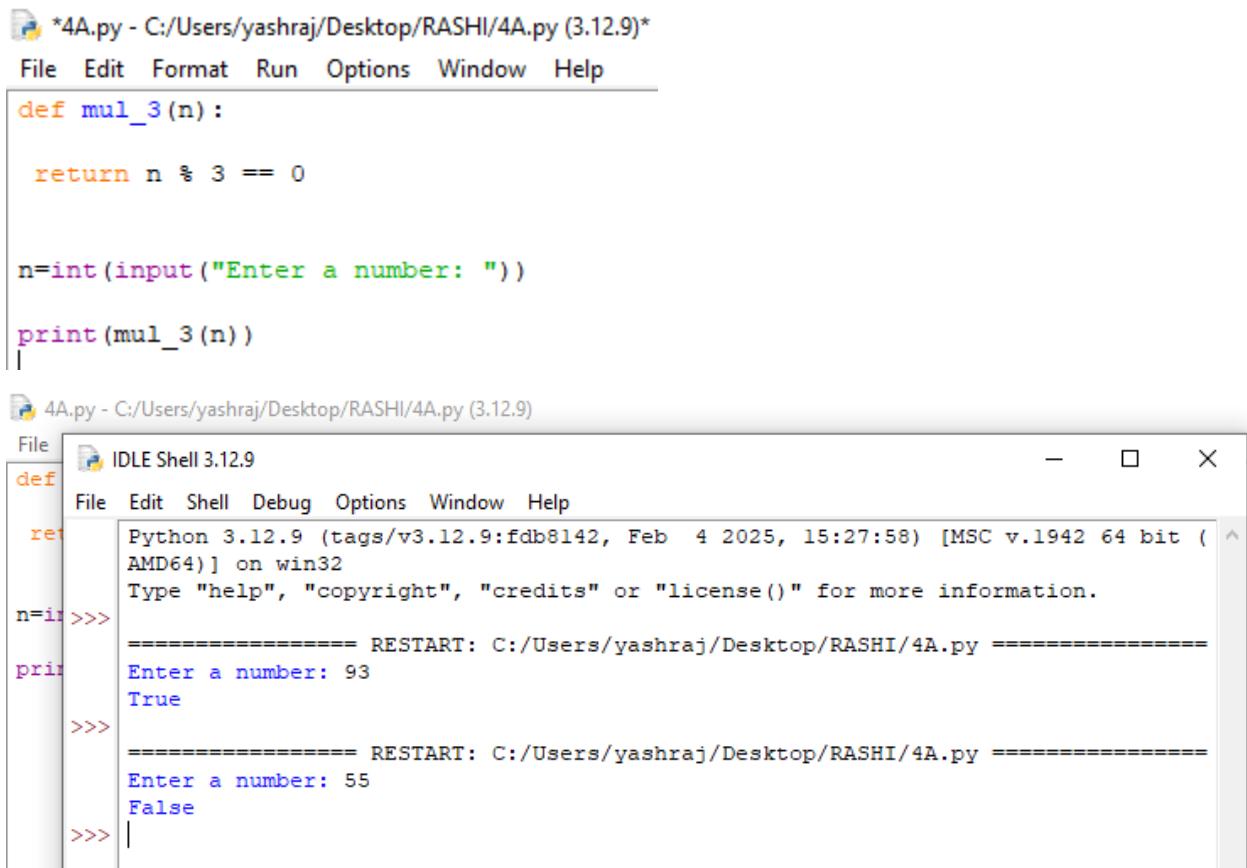
**User Defined function**

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:



The image shows a screenshot of a Python development environment. At the top, there is a file browser window titled "4A.py - C:/Users/yashraj/Desktop/RASHI/4A.py (3.12.9)". Below it is a menu bar with File, Edit, Format, Run, Options, Window, and Help. The main code editor contains the following Python script:

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

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

Below the code editor is an "IDLE Shell 3.12.9" window. It has its own menu bar with File, Edit, Shell, Debug, Options, Window, and Help. The shell displays the following interaction:

```
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:

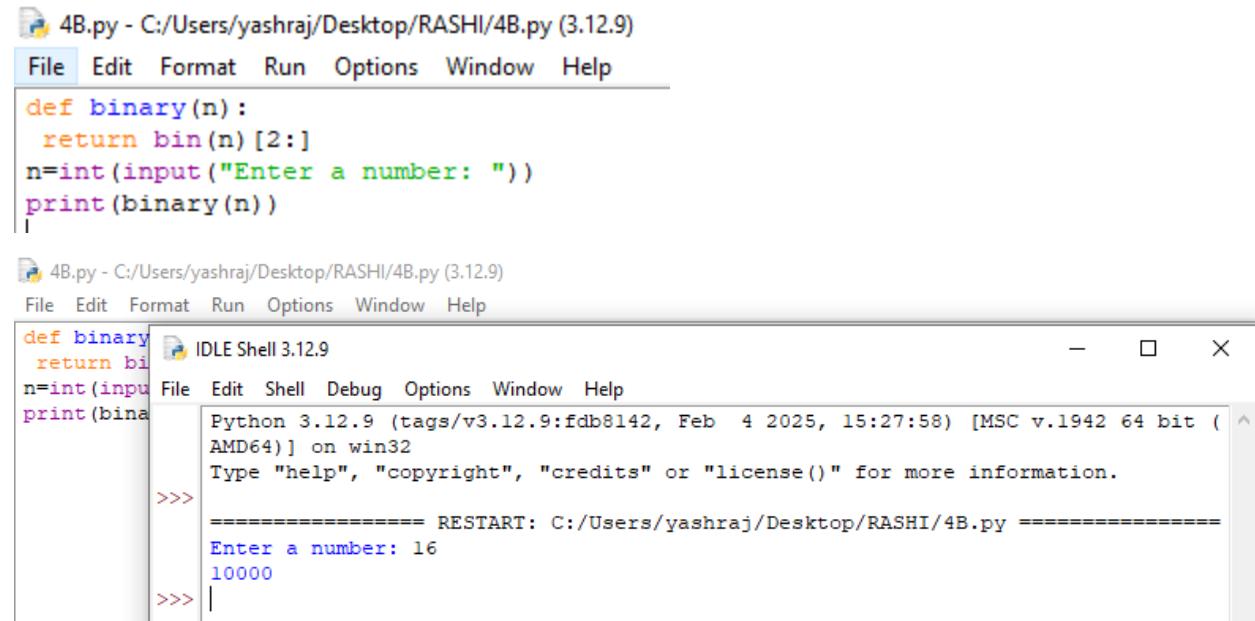
#### **Keyword as Argument**

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 shows the Python IDLE environment. At the top, there are two tabs: "4B.py - C:/Users/yashraj/Desktop/RASHI/4B.py (3.12.9)" and "4B.py - C:/Users/yashraj/Desktop/RASHI/4B.py (3.12.9)". Below the tabs is a menu bar with File, Edit, Format, Run, Options, Window, and Help. The main window contains the Python code:

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

On the right side of the window, the IDLE Shell 3.12.9 is open, showing the Python interpreter's prompt (">>>>"). It displays the following 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/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:

The screenshot shows two windows side-by-side. The left window is a code editor titled '4C.py - C:/Users/yashraj/Desktop/RASHI/4C.py (3.12.9)' containing Python code to check if a number is Armstrong. The right window is an IDLE Shell titled 'IDLE Shell 3.12.9' showing the execution of the code and its output.

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

```
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:

The screenshot shows two windows from the Python IDLE environment. The top window is a code editor with the file name '4D.py' containing the following code:

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

The bottom window is a terminal window titled 'IDLE Shell 3.12.9' showing the execution of the program. It displays the Python version information, a help message, and the output of the factorial function for the input value 6.

```
def fact(n)
    if n==0:
        return 1
    return n*
n=int(input("Enter a number: "))
print(fact
>>> ===== 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 a Windows desktop environment. In the top-left corner, there is a small icon of a document with a blue 'py' extension. Next to it, the text "5A.py - C:/Users/yashraj/Desktop/RASHI/5A.py (3.12.9)" is displayed. Below this, a standard Windows-style menu bar is visible with options: File, Edit, Format, Run, Options, Window, Help. The main body of the window contains Python code:

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

A vertical scroll bar is located on the right side of the code area.

Below this window, another window titled "5A.py - C:/Users/yashraj/Desktop/RASHI/5A.py (3.12.9)" is open. It also has a menu bar: File, Edit, Format, Run, Options, Window, Help. The title bar also includes the file path and version information. The main area is labeled "IDLE Shell 3.12.9". It shows the same Python code as the first window. To the right of the code, a terminal window displays the output of running the script:

```
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
>>>
```

The terminal window has its own menu bar: File, Edit, Shell, Debug, Options, Window, Help. A vertical scroll bar is on the right side of the terminal window.

## Conclusion:

The code is executed successfully.

## B.

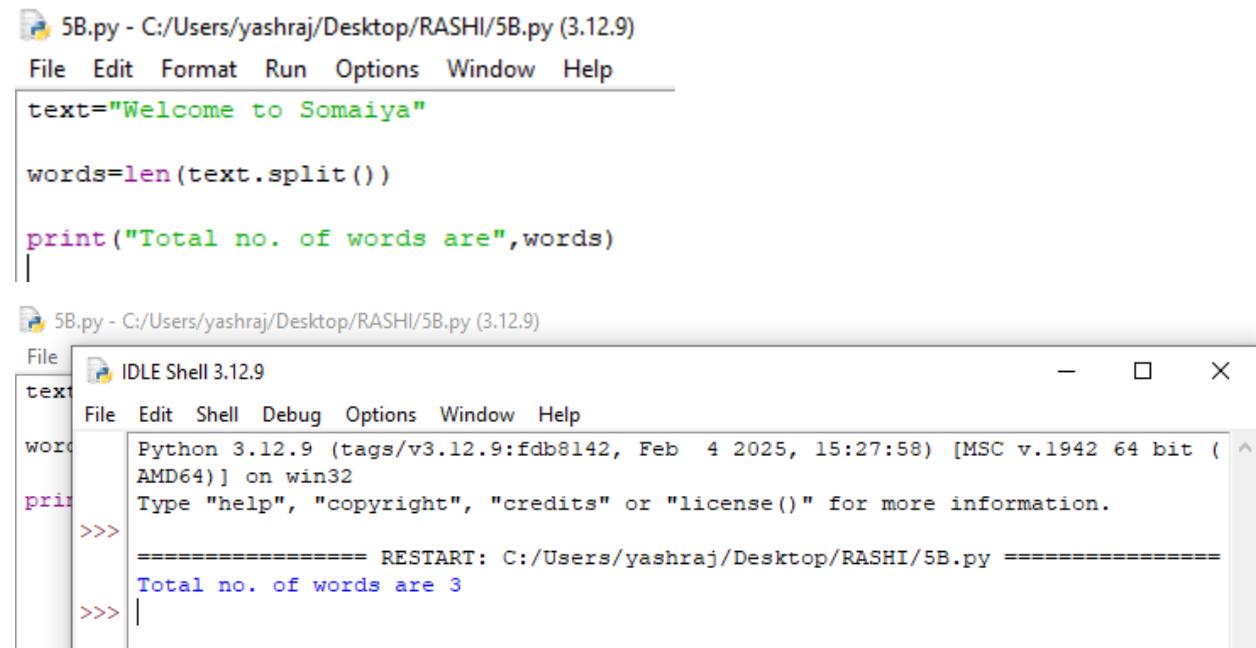
### Aim:

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 shows two windows from a Python IDE. The top window is titled '5B.py - C:/Users/yashraj/Desktop/RASHI/5B.py (3.12.9)' and contains the source code:

```
text="Welcome to Somaiya"  
  
words=len(text.split())  
  
print("Total no. of words are",words)
```

The bottom window is titled 'IDLE Shell 3.12.9' and shows the execution of the code:

```
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/5B.py ======  
>>> Total no. of words are 3  
>>> |
```

### 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:**

The screenshot shows a Python code editor and a terminal window. The code editor displays the following Python script:

```

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)

```

The terminal window below it shows the execution of the script and its 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)

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.

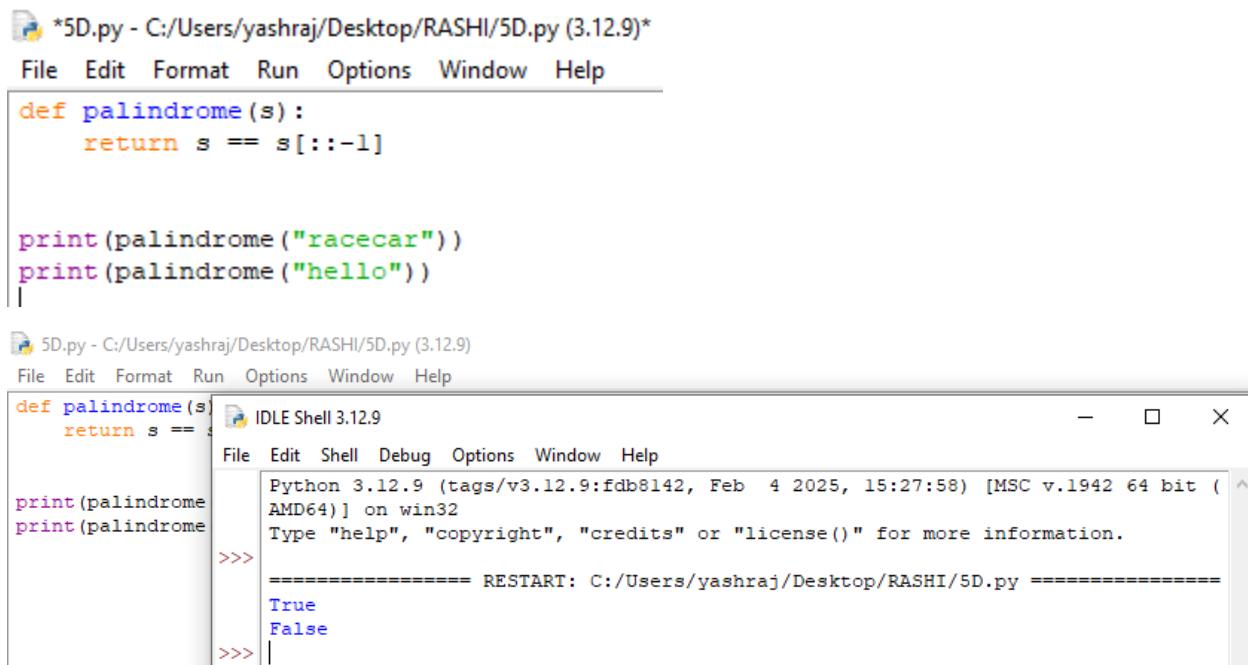
### Aim:

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 shows a Python IDE interface with two windows. The top window is a code editor titled '5D.py - C:/Users/yashraj/Desktop/RASHI/5D.py (3.12.9)\*' containing the provided Python code. The bottom window is the 'IDLE Shell 3.12.9' window, which displays the output of running the script. The shell window shows the Python interpreter's prompt (>>>), the command 'print(palindrome("racecar"))', the response 'True', the command 'print(palindrome("hello"))', and the response 'False'.

```
*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]
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:

 6A.py - C:/Users/yashraj/Desktop/RASHI/6A.py (3.12.9)

File Edit Format Run Options Window Help

```
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))
```

 6A.py - C:/Users/yashraj/Desktop/RASHI/6A.py (3.12.9)

File Edit Format Run Options Window Help

 IDLE Shell 3.12.9

File Edit Shell Debug Options Window Help

```
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))  
  
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/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 10elements, 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.

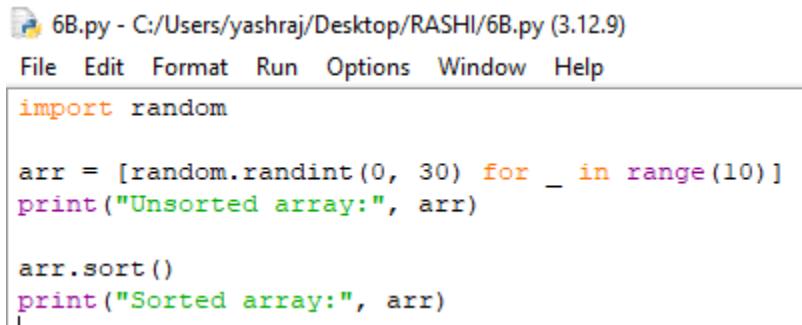
### **Source Code:**

```
import random
```

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

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

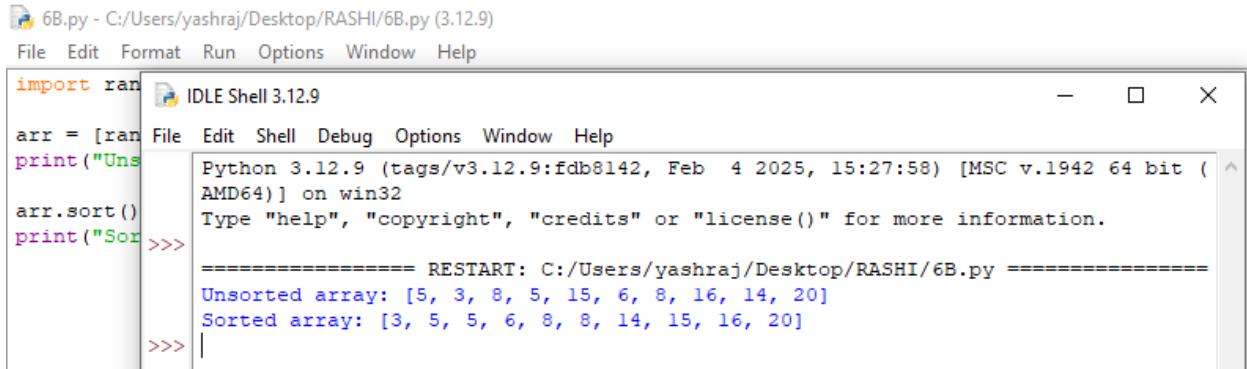
## Output:



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

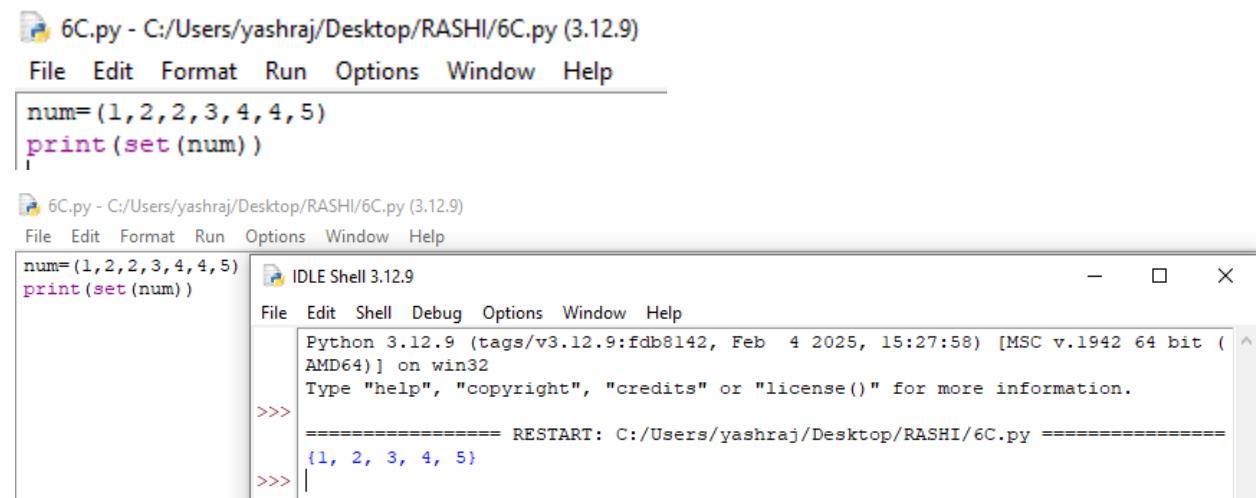
### Aim:

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 shows the Python IDLE environment. On the left, there is a code editor window titled "6C.py - C:/Users/yashraj/Desktop/RASHI/6C.py (3.12.9)" containing the following Python code:

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

To the right of the code editor is a terminal window titled "IDLE Shell 3.12.9" with the Python interpreter running. The terminal output shows:

```
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.

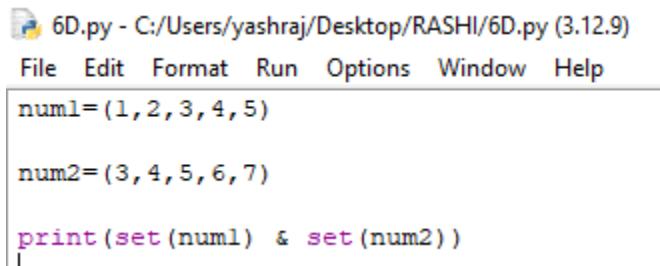
### Source Code:

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

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

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

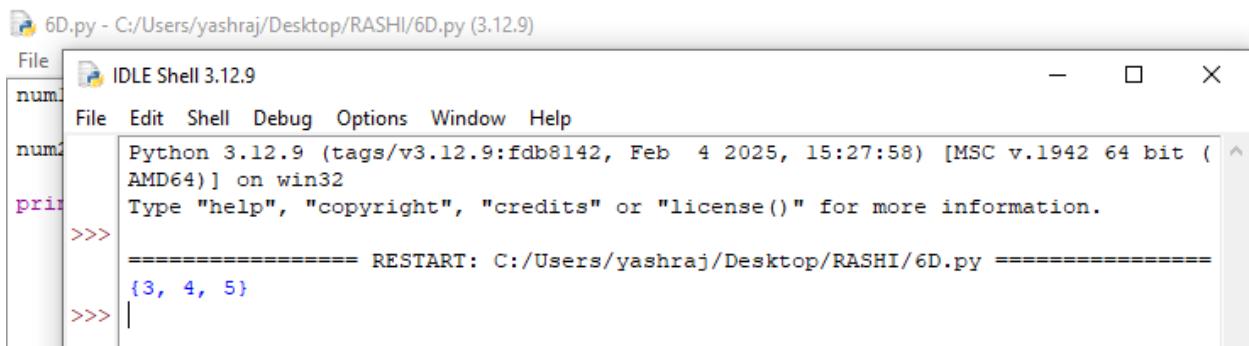
### Output:



```
6D.py - C:/Users/yashraj/Desktop/RASHI/6D.py (3.12.9)
File Edit Format Run Options Window Help
num1=(1, 2, 3, 4, 5)

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

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



```
6D.py - C:/Users/yashraj/Desktop/RASHI/6D.py (3.12.9)
File 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/6D.py =====
{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=()
IDLE Shell 3.12.9
print(empty_tuple)
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/7A.py =====
()
()
(1, 2, 3)
(1, 'Hello', 3.14, True)
((1, 2, 3), ('A', 'B', 'C'))
Middle Element: 3.14
print("Mid")
```

## 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)           IDLE Shell 3.12.9
tuple2=("a","b","c")       File Edit Shell Debug Options Window Help
joined_tuple=tuple1+tuple2 Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
print("Joined Tuple:",joined_tuple) 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"}
```

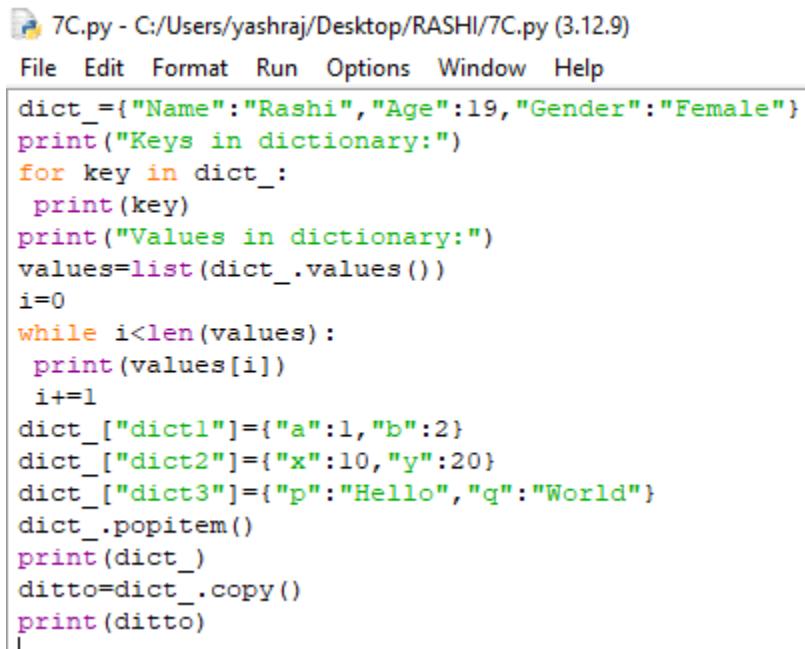
```

dict_.popitem()
print(dict_)

ditto=dict_.copy()
print(ditto)

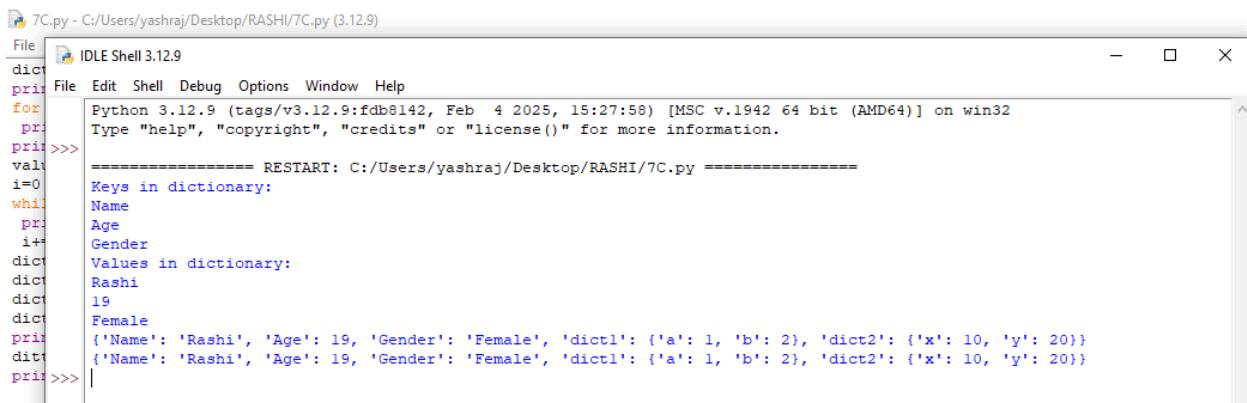
```

## Output:



```

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



```

7C.py - C:/Users/yashraj/Desktop/RASHI/7C.py (3.12.9)
File IDLE Shell 3.12.9
dict_
print File Edit Shell Debug Options Window Help
for Python 3.12.9 (tags/v3.12.9:fdb8142, Feb 4 2025, 15:27:58) [MSC v.1942 64 bit (AMD64)] on win32
pr: Type "help", "copyright", "credits" or "license()" for more information.
print >>>
val ===== RESTART: C:/Users/yashraj/Desktop/RASHI/7C.py =====
i=0
whi
pr:
val
=====
Keys in dictionary:
Name
Age
Gender
dict_ Values in dictionary:
dict_ Rashi
dict_ 19
dict_ Female
print {'Name': 'Rashi', 'Age': 19, 'Gender': 'Female', 'dict1': {'a': 1, 'b': 2}, 'dict2': {'x': 10, 'y': 20}}
dict_ {'Name': 'Rashi', 'Age': 19, 'Gender': 'Female', 'dict1': {'a': 1, 'b': 2}, 'dict2': {'x': 10, 'y': 20}}
print >>>
```

## Conclusion:

The code is executed successfully.

## D.

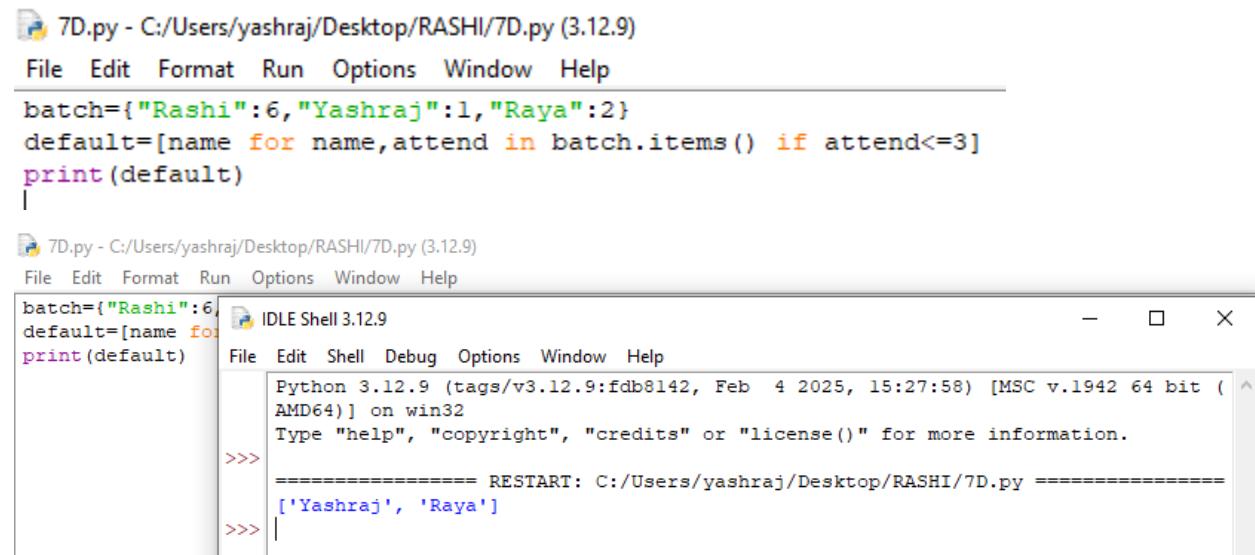
### Aim:

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:



The screenshot shows a Python IDE interface with two windows. The left window is a code editor titled '7D.py - C:/Users/yashraj/Desktop/RASHI/7D.py (3.12.9)' containing the provided Python code. The right window is an 'IDLE Shell 3.12.9' window titled 'IDLE Shell 3.12.9' showing the execution of the code. The shell output includes the Python version information ('Python 3.12.9'), the restart message ('RESTART: C:/Users/yashraj/Desktop/RASHI/7D.py'), and the resulting list of defaulters ('['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, rt, wt, at, 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
```

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/8A.py =====
```

```
File content (R mode)
x, a, r, w, rt, wt, at, 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
```

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
```

py - Notepad

```
File Edit Format View Help
```

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

**Conclusion:**

The code is executed successfully.

## **B.**

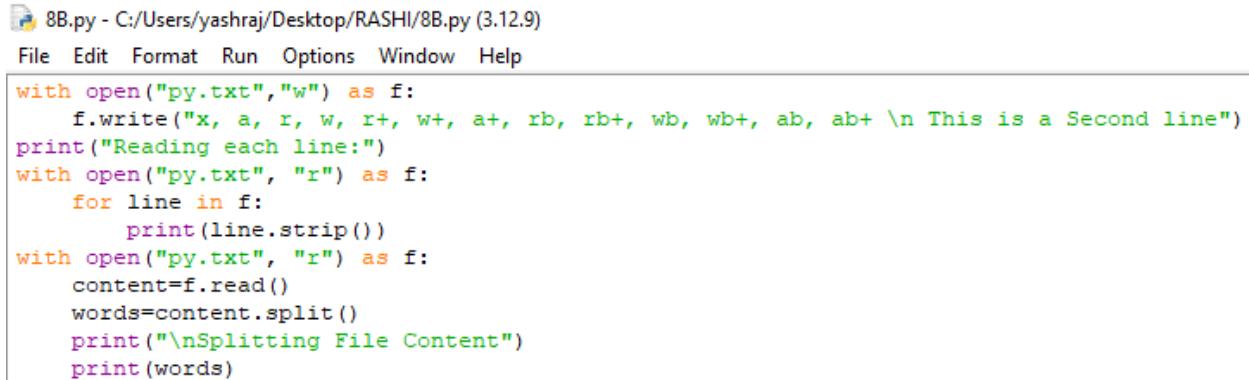
### **Aim:**

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.

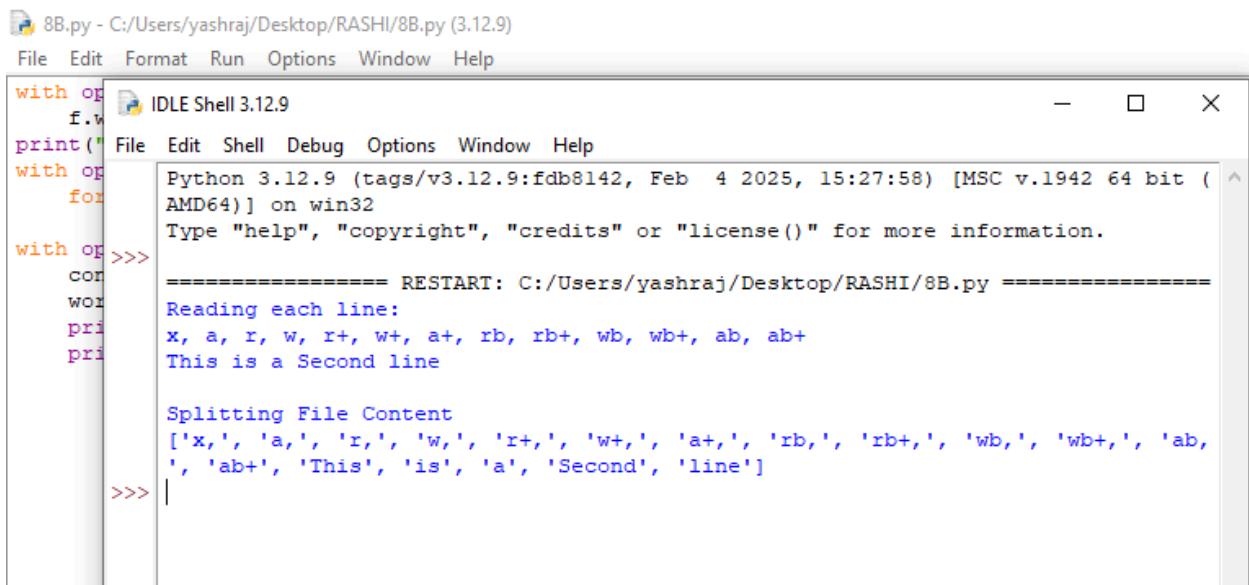
### **Source Code:**

```
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, rt, wt, at, 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)
```



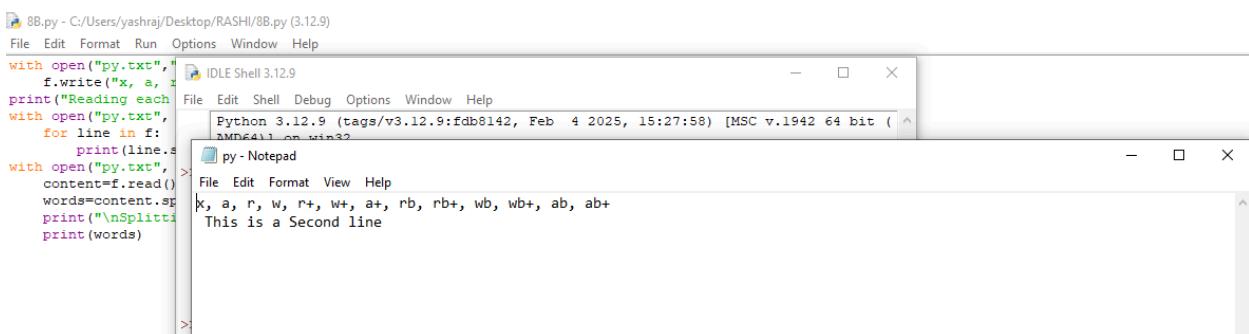
```
8B.py - C:/Users/yashraj/Desktop/RASHI/8B.py (3.12.9)
File Edit Format Run Options Window Help
with op
f.w
print('
with op
for
with op
>>>
con
wor
pri
pri
Splitting File Content
['x,', 'a,', 'r,', 'w,', 'rt,', 'wt,', 'at,', 'rb,', 'rb+', 'wb,', 'wb+', 'ab,
', 'ab+', 'This', 'is', 'a', 'Second', 'line']
>>> |
```

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/8B.py =====
Reading each line:
x, a, r, w, rt, wt, at, rb, rb+, wb, wb+, ab, ab+
This is a Second line

Splitting File Content
['x,', 'a,', 'r,', 'w,', 'rt,', 'wt,', 'at,', '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, r, w, rt, wt, at, rb, rb+, wb, wb+, ab, ab+
print("Reading each line:")
with open("py.txt",
    for line in f:
        print(line.strip())
with open("py.txt",
>>>
content=f.read()
words=content.split()
print("\nSplitting File Content")
print(words)
```

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/8B.py =====
Reading each line:
x, a, r, w, rt, wt, at, rb, rb+, wb, wb+, ab, ab+
This is a Second line
```

py - Notepad

```
File Edit Format View Help
File Edit Format View Help
x, a, r, w, rt, wt, at, 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():  
        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 di
se
se
def di
pr>>>
pr>>>
pr>>>
obj = Pers
obj.displa
>>> |
```

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/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

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/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 a screenshot of a Windows desktop environment. At the top, there is a taskbar with several pinned icons. Below the taskbar, there is a system tray showing battery status and other icons.

The main area of the screen displays two windows:

- Code Editor Window:** The title bar says "10A.py - C:/Users/yashraj/Desktop/RASHI/10A.py (3.12.9)". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code itself is as follows:

```
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()
```
- IDLE Shell 3.12.9 Window:** The title bar says "IDLE Shell 3.12.9". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The shell window shows the Python interpreter's prompt and the output of running the script:

```
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* ot *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 Stud
    def __
    se
    se
    se
    def ch>>>
        if
        el
    def ch>>>
        if
        el
    def we
        pr
student1=S
student1.w
student1.c
student1.q
```

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/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")

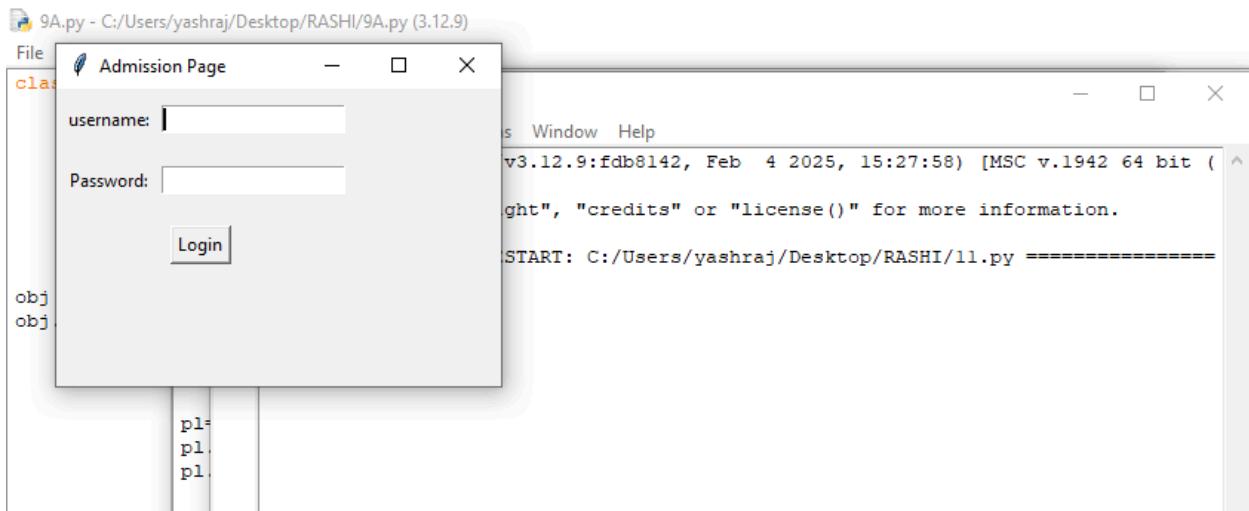
pl=tk.Tk()
pl.title("Admission Page")
pl.geometry("300x200")

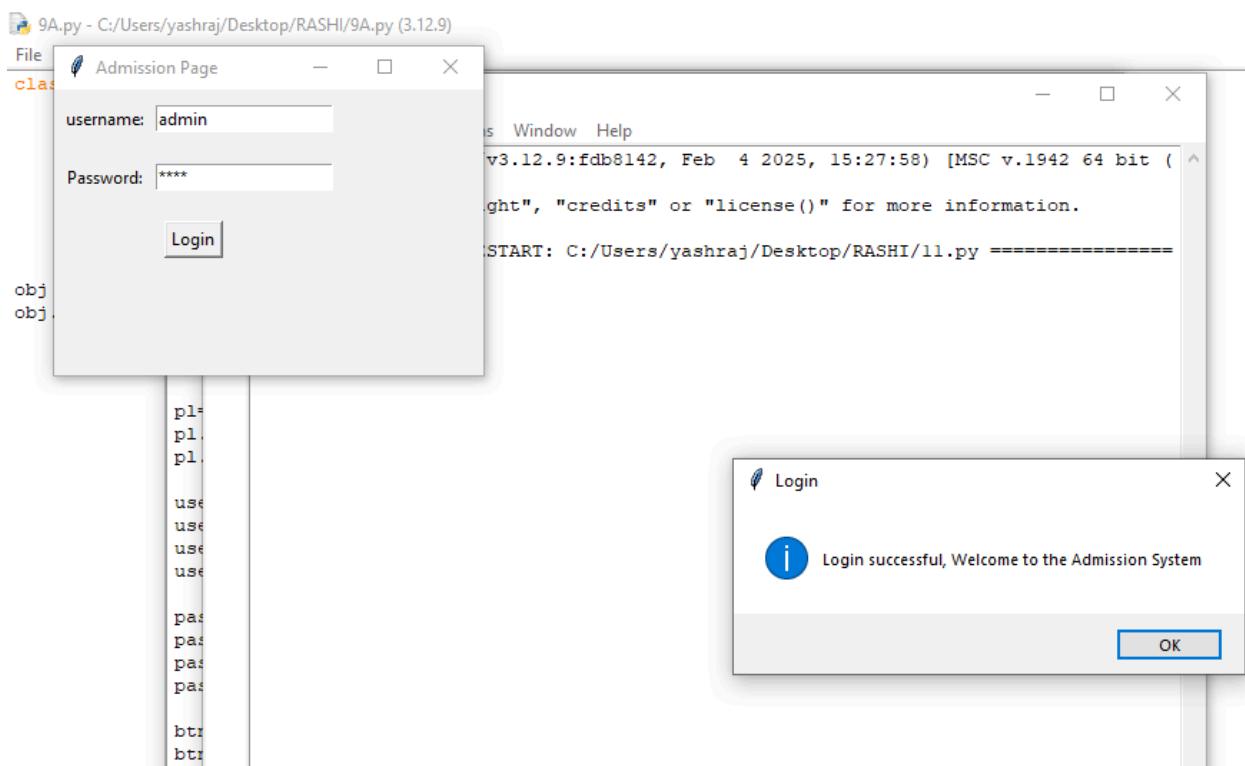
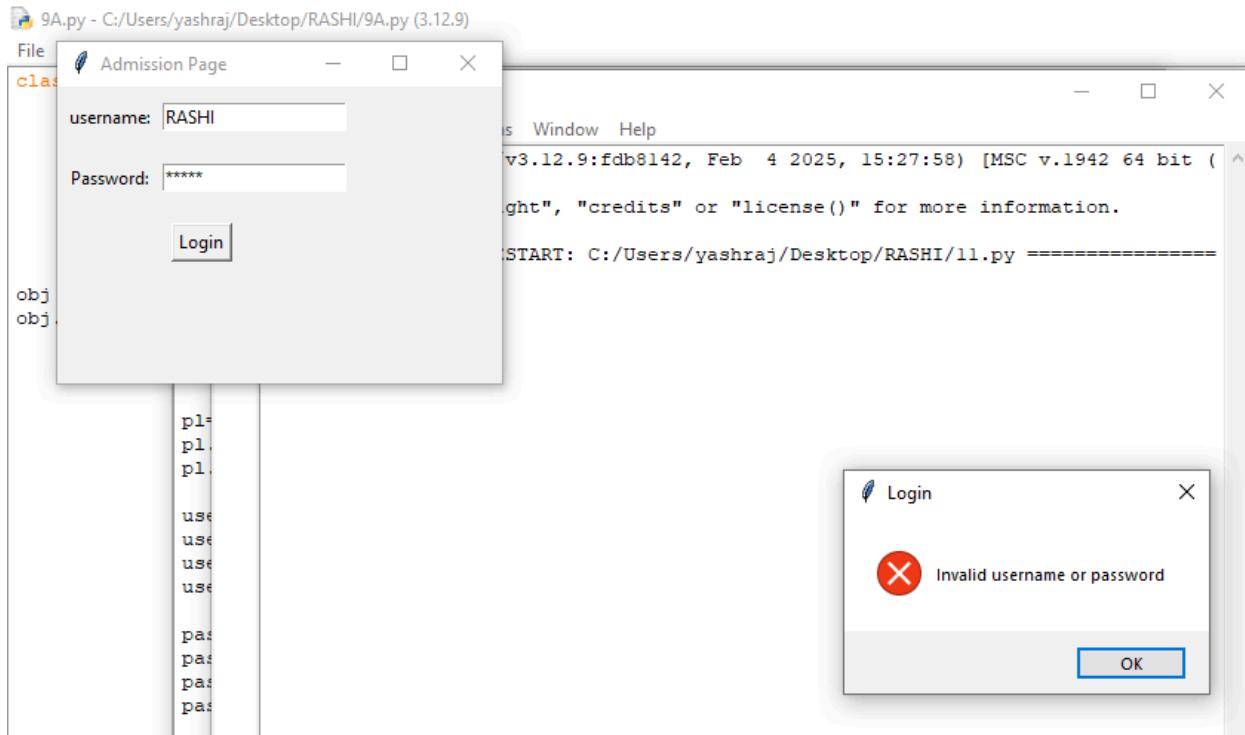
user_label=tk.Label(pl,text="username:")
user_label.grid(row=0,column=0,pady=10,padx=5)
user_entry=tk.Entry(pl)
user_entry.grid(row=0,column=1)

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()
```





## 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)
```

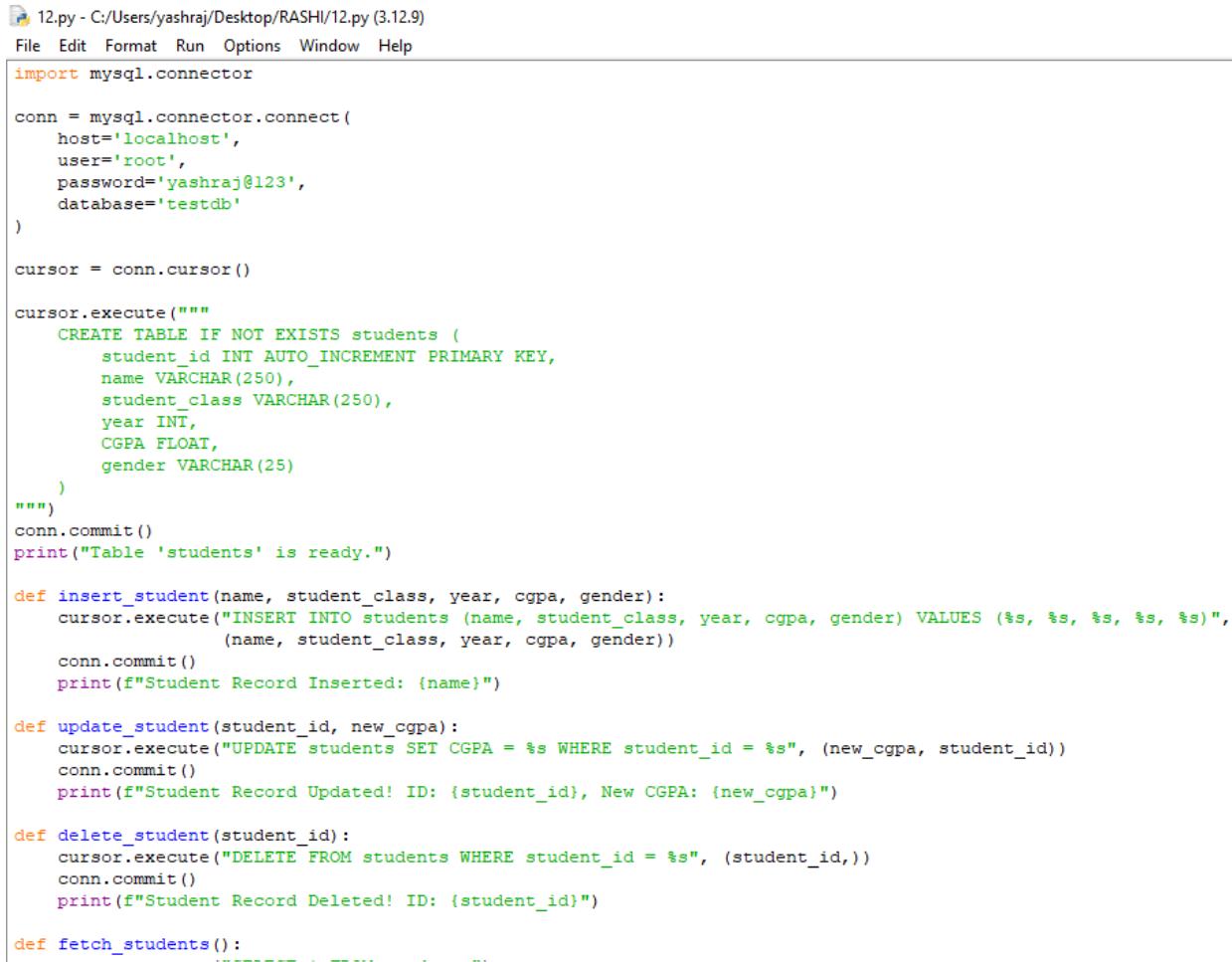
```
fetch_students()
```

```
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)

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.

```

The screenshot shows two windows side-by-side. The left window is an IDLE Shell 3.12.9 session displaying Python code for creating a MySQL database and table, and performing basic operations like insert, update, and delete. The right window is a MySQL 8.0 Command Line Client showing the results of these operations.

```

import mysql.connector
conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='password',
    database='testdb'
)
cursor = conn.cursor()
cursor.execute('CREATE TABLE students (student_id int, name varchar(250), student_class varchar(250), year int, CGPA float, gender varchar(25))')
conn.commit()
print("Table created successfully")
def insert_st():
    cursor.execute('INSERT INTO students VALUES (1, "Raya", "SYCS", 2024, 9.85, "Female")')
    conn.commit()
    print(f"1 row inserted successfully")
def update_st():
    cursor.execute('UPDATE students SET CGPA = 9.85 WHERE name = "Raya"')
    conn.commit()
    print(f"1 row updated successfully")
def delete_st():
    cursor.execute('DELETE FROM students WHERE name = "Raya"')
    conn.commit()
    print(f"1 row deleted successfully")

```

The MySQL command line client output shows:

```

MySQL 8.0 Command Line Client
Copyright (c) 2023, Oracle and/or its affiliates. All rights reserved.
This software is licensed under the MySQL Community Server License Agreement.
Distribution or transmission to third parties is prohibited without written permission.
Use is subject to license terms.

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

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.

