

# PRACTICAL NO. 3:

## WRITE SIMPLE PYTHON PROGRAM USING OPERATORS: ARITHMETIC OPERATORS, LOGICAL OPERATORS, BITWISE OPERATORS

---

### ARITHMETIC OPERATORS

---

These operators are used to perform mathematical operations and are often known as arithmetical operators. They include addition, subtraction, division, multiplication, floor division and exponential calculation. They are binary operators except for increment/decrement operators that are otherwise unary

#### EXAMPLE

```
'''PRACTICAL NO. 3: WRITE SIMPLE PYTHON PROGRAM USING OPERATORS: ARITHMETIC  
OPERATORS, LOGICAL OPERATORS, BITWISE OPERATORS'''  
  
# ARITHMETIC OPERATORS  
  
a = 10  
b = 5  
  
print("Addition is : ", (a + b))  
print("Subtraction is : ", (a - b))  
print("Multiplication is : ", (a * b))  
print("Division is : ", (a / b))  
print("Exponential is : ", (a ** b))  
print("Floor Division is : ", (a // b))  
  
main x  
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmP  
Addition is : 15  
Subtraction is : 5  
Multiplication is : 50  
Division is : 2.0  
Exponential is : 9765625  
Floor Division is : 2
```

---

## LOGICAL OPERATORS

---

These operators are used to combine relational operators. According to the evaluation of their expression, their result is in true or false form

Logical operators include AND, OR and NOT

### EXAMPLE

```
# LOGICAL OPERATORS
|
a = True
b = False
c = True
print("Logical AND : True and False ",(a and b))
print("Logical AND : True and True ",(a and c))
print("Logical OR : True or False ",(a or b))
print("Logical OR : True or True ",(a or c))
print("Logical NOT : not True ",(not a))
print("Logical NOT: not False ",(not b))

main x
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects
Logical AND : True and False  False
Logical AND : True and True  True
Logical OR : True or False  True
Logical OR : True or True  True
Logical NOT :  not True  False
Logical NOT: not False  True

Process finished with exit code 0
```

---

## BITWISE OPERATORS

---

They are used to perform bit by bit operation. They can be both unary and binary. First the number is converted into bits and then operation is directly performed upon them .They include Bitwise AND, Bitwise OR and Bitwise NOT, Bitwise left shift and Right shift

```
# BITWISE OPERATORS

x = 10
y = 4

print("Bitwise AND : ", (x & y))
print("Bitwise OR : ", (x | y))
print("Bitwise NOT: ", (~x))
print("Bitwise XOR : ", (x ^ y))
print("Bitwise RIGHT SHIFT : ", (x >> y))
print("Bitwise LEFT SHIFT : ", (x << y))

main x
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/pythonProject/main.py
Bitwise AND :  0
Bitwise OR :  14
Bitwise NOT: -11
Bitwise XOR :  14
Bitwise RIGHT SHIFT :  0
Bitwise LEFT SHIFT :  160

Process finished with exit code 0
```

---

## ASSIGNMENT OPERATORS

---

These operators are used to assign values to variables. The value to be assigned is at the right side whereas the variable to which value should be assigned is at the left side

```
#ASSIGNMENT OPERATORS

x=5
print(x)
x += 5
print(x)
x -= 5
print(x)
x *= 5
print(x)
x %= 5
print(x)
x /= 5
print(x)
x //= 10
print(x)
x **= 5
print(x)
```

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts'
5
10
5
25
0
1.0
0.0
0.0

Process finished with exit code 0
```

---

## IDENTITY OPERATORS

---

These operators are used to compare memory address of two objects.

They are used to check whether two values are located at the same part of memory or not

```
# SPECIAL OPERATORS
# IDENTITY OPERATORS

x1 = 5
y1 = 5
x2 = 'Hello'
y2 = 'Hello'
x3 = [1, 2, 3]
y3 = [1, 2, 3]

print(x1 is not y1)
print(x2 is y2)
print(x3 is y3)
```

```
main x
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/p
False
True
False

Process finished with exit code 0
```

## MEMBERSHIP OPERATORS

These operators are used to check existence of a particular element in a sequence

The result is in a Boolean form and the return values like true or false

We can use them in sequences like string, tuple, string, dictionary

It reduces effort of searching an element in the list

They are ‘in’ and ‘not in’

Operator	Meaning	Example
in	True if value/variable is found in the sequence	5 in x
not in	True if value/variable is not found in the sequence	5 not in x

```
#MEMBERSHIP OPERATORS

x = 'Hello world'
y = {1:'a',2:'b'}
print('H' in x)
print('hello' not in x)
print(1 in y)
print('a' in y)

main ×
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProje
True
True
True
False

Process finished with exit code 0
```

---

## COMPARISON OPERATORS

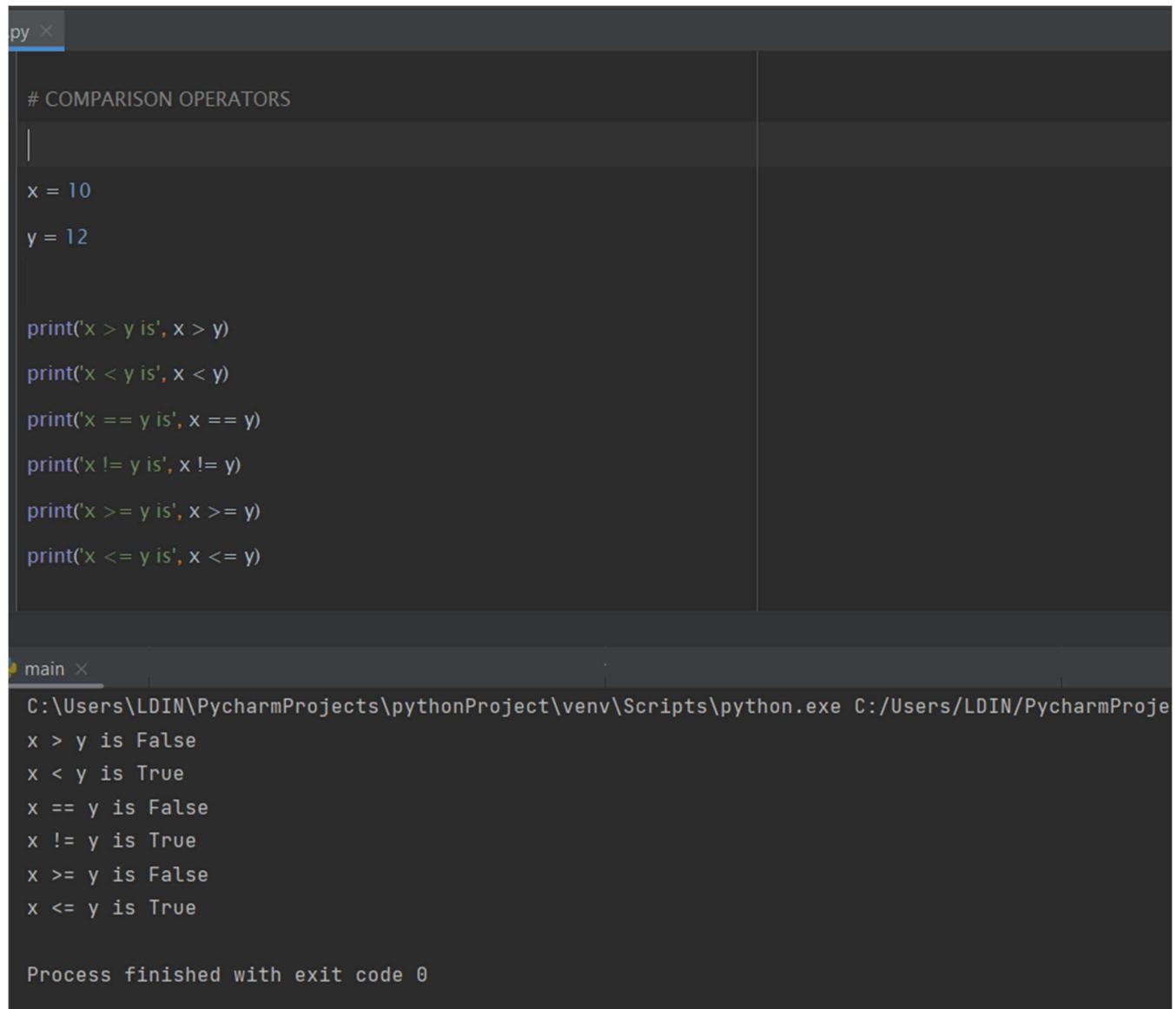
---

Also known as Relational Operators, they are used to compare values

According to the evaluation of their expression, the results are returned in a Boolean form as either True or False

They are binary operators meaning they perform operation on binary operands

They include greater than, greater than or equal to, less than, less than or equal to, equal to and not Equal to



The screenshot shows the PyCharm IDE interface. The top part displays a code editor with a file named 'py'. The code within the editor is as follows:

```
# COMPARISON OPERATORS
|
x = 10
y = 12

print('x > y is', x > y)
print('x < y is', x < y)
print('x == y is', x == y)
print('x != y is', x != y)
print('x >= y is', x >= y)
print('x <= y is', x <= y)
```

The bottom part shows a terminal window titled 'main' with the following output:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProje
x > y is False
x < y is True
x == y is False
x != y is True
x >= y is False
x <= y is True

Process finished with exit code 0
```

## XI EXERCISE

### 1.WRITE A PROGRAM TO CONVERT U.S. DOLLARS TO INDIAN RUPEES.

#### OUTPUT 1

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the following Python code:

```
87     d = float(input("Enter dollars:"))
88
89     r = d * 76.76
90
91     print(f'{d} dollars into indian rupees {r}')
```

The bottom window is titled "Run: main" and shows the execution results:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe
Enter dollars:1
1.0 dollars into indian rupees 76.76
Process finished with exit code 0
```

#### OUTPUT 2

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the same Python code as Output 1:

```
87     d = float(input("Enter dollars:"))
88
89     r = d * 76.76
90
91     print(f'{d} dollars into indian rupees {r}')
```

The bottom window is titled "Run: main" and shows the execution results:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe
Enter dollars:20
20.0 dollars into indian rupees 1535.2
Process finished with exit code 0
```

## 2. WRITE A PROGRAM TO CONVERT BITS TO MEGABYTES, GIGABYTES AND TERABYTES

### OUTPUT 1

The screenshot shows the PyCharm IDE interface. The top part displays the code in a file named 'main.py'. The bottom part shows the run terminal with the output of the program.

```
1 main.py ×
2 bits=float(input("Enter bits:"))
3 byte=0.125*bits
4 megabytes=0.000001*byte
5 gigabytes=0.001*megabytes
6 terabytes=0.001*gigabytes
7 print(f"\n{bits} bits are {megabytes} Megabytes ,\n {gigabytes} Gigabytes \n and {terabytes} Terabytes")
8
Run: main ×
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/pythonProject/main.py
Enter bits:12
12.0 bits are 1.5e-06 Megabytes ,
1.5e-09 Gigabytes
and 1.5e-12 Terabytes

Process finished with exit code 0
```

### OUTPUT 2

The screenshot shows the PyCharm IDE interface. The top part displays the code in a file named 'main.py'. The bottom part shows the run terminal with the output of the program.

```
1 main.py ×
2 bits=float(input("Enter bits:"))
3 byte=0.125*bits
4 megabytes=0.000001*byte
5 gigabytes=0.001*megabytes
6 terabytes=0.001*gigabytes
7 print(f"\n{bits} bits are {megabytes} Megabytes ,\n {gigabytes} Gigabytes \n and {terabytes} Terabytes")
8
Run: main ×
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/pythonProject/main.py
Enter bits:24
24.0 bits are 3e-06 Megabytes ,
3e-09 Gigabytes
and 3e-12 Terabytes

Process finished with exit code 0
```

### 3. WRITE A PROGRAM TO FIND THE SQUARE ROOT OF A NUMBER

#### OUTPUT 1

The screenshot shows the PyCharm IDE interface. The top window displays the Python script `main.py` with the following code:

```
1
2     n = float(input('Enter a number: '))
3
4     ns = n ** 0.5
5
6     print(f'The square root of {n} is {ns}')
```

The bottom window shows the run output:

```
Run: main
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:
Enter a number: 144
The square root of 144.0 is 12.0
Process finished with exit code 0
```

#### OUTPUT 2

The screenshot shows the PyCharm IDE interface. The top window displays the Python script `main.py` with the same code as Output 1:

```
1
2     n = float(input('Enter a number: '))
3
4     ns = n ** 0.5
5
6     print(f'The square root of {n} is {ns}')
```

The bottom window shows the run output:

```
Run: main
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:
Enter a number: 196
The square root of 196.0 is 14.0
Process finished with exit code 0
```

#### 4. WRITE A PROGRAM TO FIND THE AREA OF RECTANGLE

##### OUTPUT 1

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the following Python code:

```
l = float(input('Enter the length of Rectangle : '))
b = float(input('Enter the breadth of Rectangle : '))
Area = l * b
print(f"Area of Rectangle : {Area} sq.cm")
```

The bottom window is titled "Run: main" and shows the execution results:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe
Enter the length of Rectangle : 12
Enter the breadth of Rectangle : 4
Area of Rectangle : 48.0 sq.cm
Process finished with exit code 0
```

##### OUTPUT 2

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the same Python code as in Output 1:

```
l = float(input('Enter the length of Rectangle : '))
b = float(input('Enter the breadth of Rectangle : '))
Area = l * b
print(f"Area of Rectangle : {Area} sq.cm")
```

The bottom window is titled "Run: main" and shows the execution results:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe
Enter the length of Rectangle : 10
Enter the breadth of Rectangle : 45.9
Area of Rectangle : 459.0 sq.cm
Process finished with exit code 0
```

## 5. WRITE A PROGRAM TO CALCULATE AREA AND PERIMETER OF THE SQUARE

### OUTPUT 1

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the following Python code:

```
s = int(input("Enter the Side Length of Square:"))
p = 4*s
area=s*s
print(f'Perimeter of Square: {p} cm')
print(f'Area of Square: {area} sq.cm')
```

The bottom window is titled "Run: main" and shows the execution results:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe
Enter the Side Length of Square:56
Perimeter of Square: 224 cm
Area of Square: 3136 sq.cm
Process finished with exit code 0
```

### OUTPUT 2

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the same Python code as in Output 1:

```
s = int(input('Enter the Side Length of Square:'))
p = 4*s
area=s*s
print(f'Perimeter of Square: {p} cm')
print(f'Area of Square: {area} sq.cm')
```

The bottom window is titled "Run: main" and shows the execution results:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe
Enter the Side Length of Square:54
Perimeter of Square: 216 cm
Area of Square: 2916 sq.cm
Process finished with exit code 0
```

## 6.WRITE A PROGRAM TO CALCULATE SURFACE VOLUME AND AREA OF A CYLINDER.

### OUTPUT 1

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the following Python code:

```
1 height = float(input('Height of cylinder: '))
2 radian = float(input('Radius of cylinder: '))
3 pi=3.14
4 volume = pi * radian * radian * height
5 su = ((2*pi*radian) * height) + ((pi*radian**2)*2)
6 print(f'Volume of Cylinder is: {volume}')
7 print(f'Surface Area of Cylinder is: {su}')
8
```

The bottom window is titled "Run: main" and shows the terminal output of the program. It prompts for height and radius, then prints the calculated volume and surface area.

```
Run: main
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/
Height of cylinder: 12
Radius of cylinder: 4
Volume of Cylinder is: 602.88
Surface Area of Cylinder is: 401.92
Process finished with exit code 0
```

### OUTPUT 2

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the same Python code as in Output 1.

```
1 height = float(input('Height of cylinder: '))
2 radian = float(input('Radius of cylinder: '))
3 pi=3.14
4 volume = pi * radian * radian * height
5 su = ((2*pi*radian) * height) + ((pi*radian**2)*2)
6 print(f'Volume of Cylinder is: {volume}')
7 print(f'Surface Area of Cylinder is: {su}')
8
```

The bottom window is titled "Run: main" and shows the terminal output of the program with different input values.

```
Run: main
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/
Height of cylinder: 67
Radius of cylinder: 8
Volume of Cylinder is: 13464.32
Surface Area of Cylinder is: 3768.0
Process finished with exit code 0
```

## 7. WRITE A PROGRAM TO SWAP THE VALUE OF TWO VARIABLES

### OUTPUT 1

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the following Python code:

```
1 a=int(input("enter value of a : "))
2 b=int(input("enter value of ab : "))
3 print(f'Value of variable a before swap {a} and value of variable b before swap {b}')
4 a,b=b,a
5 print(f'Value of variable a after swap {a} and value of variable b after swap {b}'')
6
```

The bottom window is titled "Run: main" and shows the terminal output:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/
enter value of a : 67
enter value of ab : 76
Value of variable a before swap 67 and value of variable b before swap 76
Value of variable a after swap 76 and value of variable b after swap 67

Process finished with exit code 0
```

### OUTPUT 2

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the same Python code as Output 1:

```
1 a=int(input("enter value of a : "))
2 b=int(input("enter value of ab : "))
3 print(f'Value of variable a before swap {a} and value of variable b before swap {b}')
4 a,b=b,a
5 print(f'Value of variable a after swap {a} and value of variable b after swap {b}'')
6
```

The bottom window is titled "Run: main" and shows the terminal output:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/
enter value of a : 12
enter value of ab : 4
Value of variable a before swap 12 and value of variable b before swap 4
Value of variable a after swap 4 and value of variable b after swap 12

Process finished with exit code 0
```

# PRACTICAL NO. 4:

## WRITE SIMPLE PYTHON PROGRAM TO DEMONSTRATE USE OF CONDITIONAL STATEMENTS: IF' STATEMENT, 'IF ... ELSE' STATEMENT, NESTED 'IF' STATEMENT

---

### IF STATEMENT

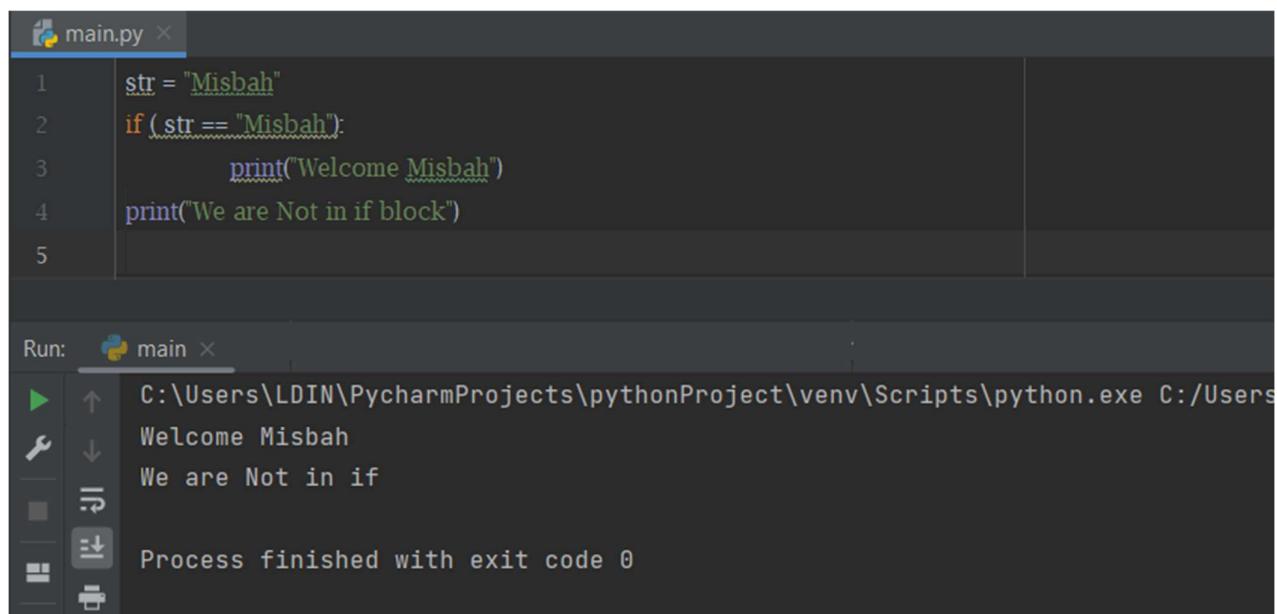
---

The if statement is the simplest decision-making statement. It is used to decide whether a certain statement or block of statements will be executed or not i.e. if a certain condition is true then a block of statement is executed otherwise not.

#### SYNTAX:

If condition:

    Statement(s)



The screenshot shows a PyCharm interface. The top window is titled 'main.py' and contains the following Python code:

```
1 str = "Misbah"
2 if (str == "Misbah"):
3     print("Welcome Misbah")
4 print("We are Not in if block")
```

The bottom window is titled 'Run' and shows the output of the program:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/ldin/PycharmProjects/pythonProject/main.py
Welcome Misbah
We are Not in if
Process finished with exit code 0
```

---

### IF-ELSE STATEMENT:

---

The if statement alone tells us that if a condition is true, it will execute a block of statements and if the condition is false, it won't. But what if we want to do something else if the condition is false. Here comes the else statement. We can use the else statement with if statement to execute a block of code when the condition is false.

## SYNTAX:

```
if (condition):
    # if Condition is true, Executes this block
else:
    # if condition is false, Executes this block
```

The screenshot shows the PyCharm IDE interface. The top window is titled 'main.py' and contains the following code:

```
1 i=10;
2 if(i<20):
3     print("i is smaller than 20")
4 else:
5     print("i is greater than 25")
6
```

The bottom window is titled 'Run' and shows the output of the script execution:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/
i is smaller than 20

Process finished with exit code 0
```

---

## NESTED-IF STATEMENT:

---

A nested if is an if statement that is the target of another if statement. Nested if statements means an if statement inside another if statement. Yes, Python allows us to nest if statements within if statements. i.e, we can place an if statement inside another if statement.

## SYNTAX:

```
if (condition1):
    # Executes when condition1 is true
    if (condition2):
        # Executes when condition2 is true
        # if Block is end here
    # if Block is end here
```

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the following Python code:

```
i = 50
if(i == 50):
    # First if statement
    if(i < 80):
        print("i is smaller than 20")
    # Nested - if statement will only be executed if statement above is true
    if(i < 95):
        print("i is smaller than 95 too")
    else:
        print("i is greater than 95")
```

The bottom window is titled "Run: main" and shows the output of the program:

```
C:/Users/LDIN/PycharmProjects/pythonProject/venv/Scripts/python.exe C:/Users/
i is smaller than 20
i is smaller than 95 too
Process finished with exit code 0
```

---

## XI. EXERCISE

---

### 1. WRITE A PROGRAM TO CHECK WHETHER A NUMBER IS EVEN OR ODD

The screenshot shows the PyCharm IDE interface. The top window is titled "main.py" and contains the following Python code:

```
num = int(input("Enter a number: "))
if (num % 2) == 0:
    print(f'{num} is Even')
else:
    print(f'{num} is Odd')
```

The bottom window is titled "Run: main" and shows the output of the program:

```
C:/Users/LDIN/PycharmProjects/pythonProject/venv/Scripts/python.exe C:/Users/LDIN/
Enter a number: 44
44 is Even
Process finished with exit code 0
```

2. WRITE A PROGRAM TO FIND OUT ABSOLUTE VALUE OF AN INPUT NUMBER

The screenshot shows the PyCharm IDE interface. The top part displays the code in a file named `main.py`:

```
main.py ×
num = float(input("Enter a number:"))
num = abs(num)
print(f"Absolute number:{num}")
```

The bottom part shows the run output for the script `main`:

```
main ×
C:/Users/LDIN/PycharmProjects/pythonProject/venv/Scripts/python.exe C:/Users/LDIN
Enter a number:-97
Absolute number:97.0

Process finished with exit code 0
```

3. WRITE A PROGRAM TO CHECK THE LARGEST NUMBER AMONG THE THREE NUMBERS

The screenshot shows the PyCharm IDE interface. The top part displays the code in a file named `main.py`:

```
main.py ×
1 n1 = float(input("Enter first number: "))
2 n2 = float(input("Enter second number: "))
3 n3 = float(input("Enter third number: "))
4 if (n1 >= n2) and (n1 >= n3):
5     print(f"The largest number is {n1}")
6 elif (n2 >= n1) and (n2 >= n3):
7     print(f"The largest number is {n2}")
8 else:
9     print(f"The largest number is {n3}")
10 |
```

The bottom part shows the run output for the script `main`:

```
Run: main ×
C:/Users/LDIN/PycharmProjects/pythonProject/venv/Scripts/python.exe C:/Users/LDIN/Pych
Enter first number: 57
Enter second number: 89
Enter third number: 91
The largest number is 91.0

Process finished with exit code 0
```

4. WRITE A PROGRAM TO CHECK IF THE INPUT YEAR IS A LEAP YEAR OR NOT

The screenshot shows the PyCharm IDE interface. The top window displays the Python script `main.py` with the following code:

```
1 year = int(input("Enter a year: "))
2 if (year % 4 == 0):
3     print(f"{year} is a leap year")
4 else:
5     print(f"{year} is not a leap year")
6
```

The bottom window shows the run output for the year 2044:

```
Run: main ×
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/pythonProject/main.py
Enter a year: 2044
2044 is a leap year
Process finished with exit code 0
```

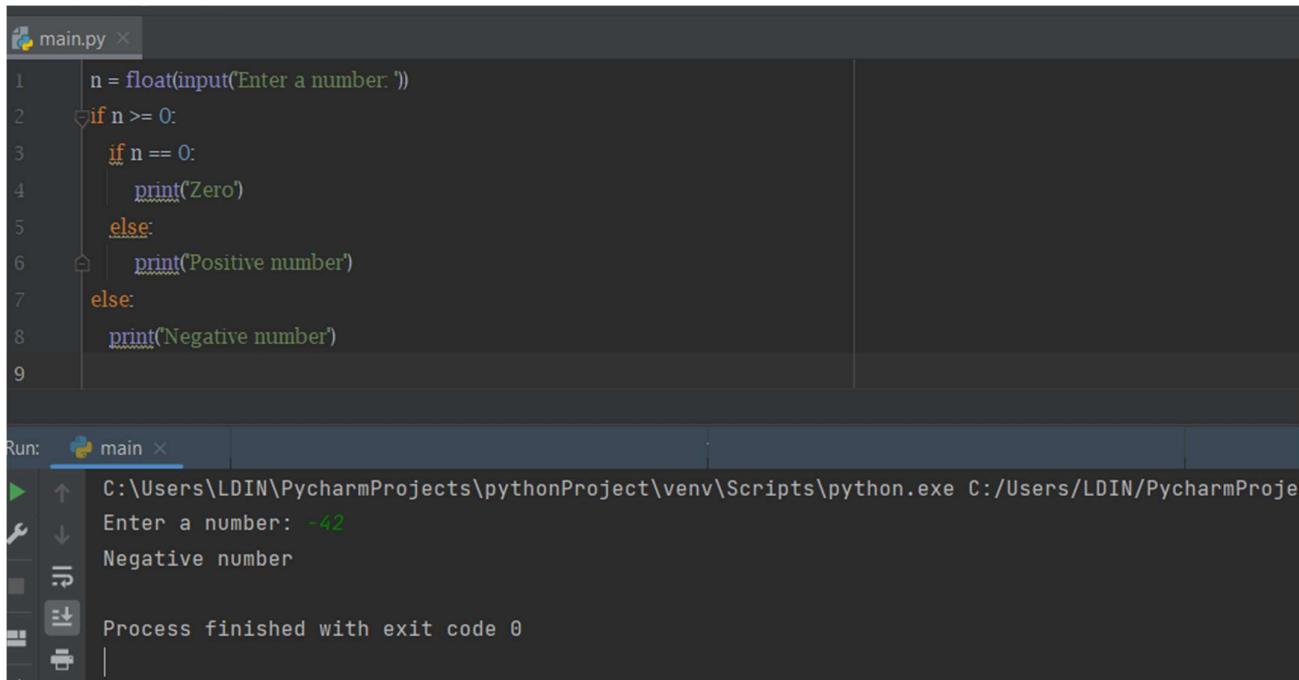
5. WRITE A PROGRAM TO CHECK IF A NUMBER IS POSITIVE, NEGATIVE OR ZERO

The screenshot shows the PyCharm IDE interface. The top window displays the Python script `main.py` with the following code:

```
1 n = float(input("Enter a number: "))
2 if n >= 0:
3     if n == 0:
4         print("Zero")
5     else:
6         print("Positive number")
7 else:
8     print("Negative number")
9
```

The bottom window shows the run output for the number 69:

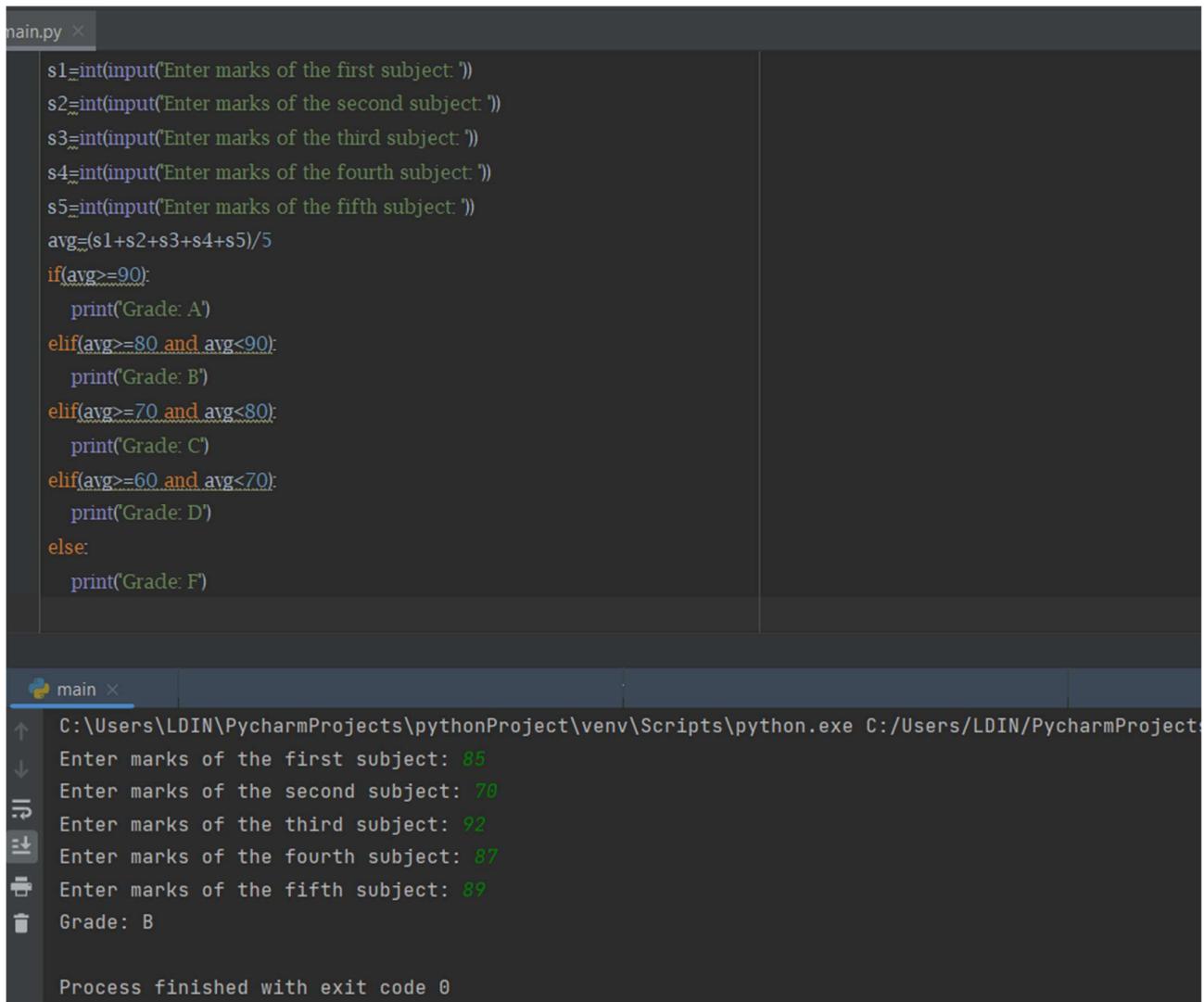
```
Run: main ×
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/pythonProject/main.py
Enter a number: 69
Positive number
Process finished with exit code 0
```



```
main.py x
1 n = float(input("Enter a number: "))
2 if n >= 0:
3     if n == 0:
4         print("Zero")
5     else:
6         print("Positive number")
7 else:
8     print("Negative number")
9
```

Run: main x  
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/main.py  
Enter a number: -42  
Negative number  
Process finished with exit code 0

6. WRITE A PROGRAM THAT TAKES THE MARKS OF 5 SUBJECTS AND DISPLAYS THE GRADE.



```
main.py x
1 s1=int(input("Enter marks of the first subject: "))
2 s2=int(input("Enter marks of the second subject: "))
3 s3=int(input("Enter marks of the third subject: "))
4 s4=int(input("Enter marks of the fourth subject: "))
5 s5=int(input("Enter marks of the fifth subject: "))
6 avg=(s1+s2+s3+s4+s5)/5
7 if(avg>=90):
8     print("Grade: A")
9 elif(avg>=80 and avg<90):
10    print("Grade: B")
11 elif(avg>=70 and avg<80):
12    print("Grade: C")
13 elif(avg>=60 and avg<70):
14    print("Grade: D")
15 else:
16    print("Grade: F")
```

Run: main x  
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/main.py  
Enter marks of the first subject: 85  
Enter marks of the second subject: 70  
Enter marks of the third subject: 92  
Enter marks of the fourth subject: 87  
Enter marks of the fifth subject: 89  
Grade: B  
Process finished with exit code 0

# PRACTICAL NO. 5:

## WRITE PYTHON PROGRAM TO DEMONSTRATE USE OF LOOPING STATEMENTS: ‘WHILE’ LOOP, ‘FOR’ LOOP AND NESTED LOOP

---

### WHILE LOOP:

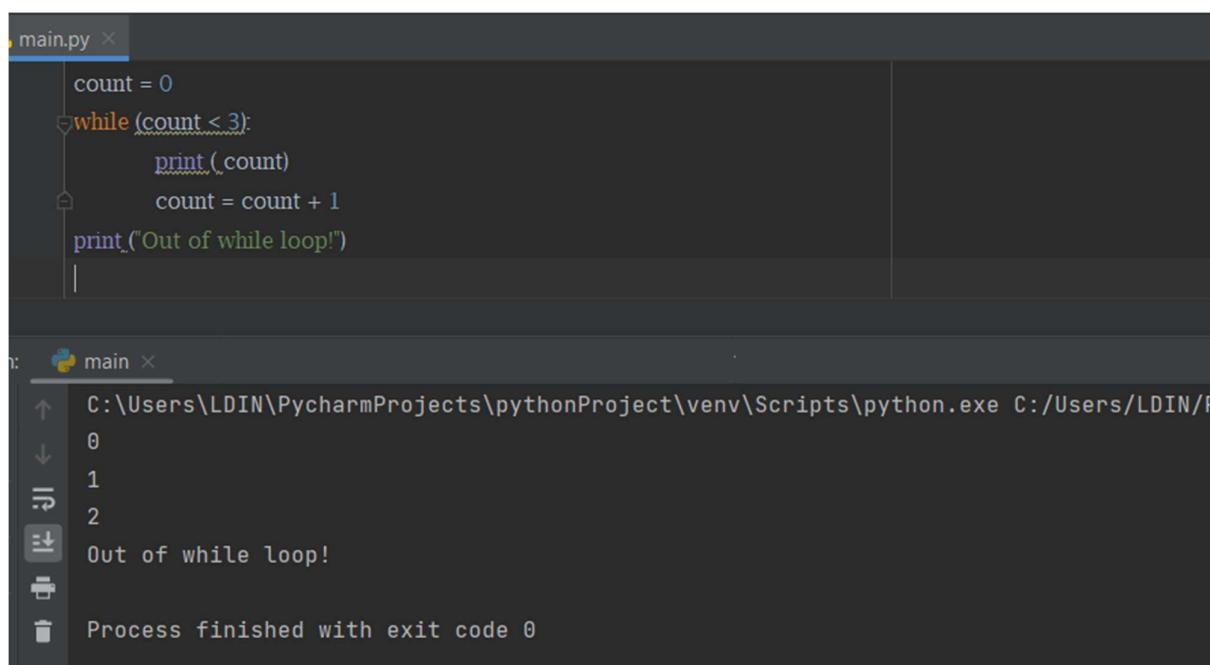
---

A **while** loop statement in Python programming language repeatedly executes a target statement as long as a given condition is true.

#### SYNTAX :

```
iteraror_var=initialized_value  
while expression:  
    statement(s)  
    iterator inc/dec
```

The loop iterates while the condition is true. When the condition becomes false, program control passes to the line immediately following the loop



The screenshot shows a PyCharm interface with two panes. The top pane displays the code file 'main.py' containing the following Python code:

```
count = 0
while (count < 3):
    print(count)
    count = count + 1
print("Out of while loop!")
```

The bottom pane shows the terminal output of running the script:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/P
0
1
2
Out of while loop!
Process finished with exit code 0
```

---

## FOR LOOP:

---

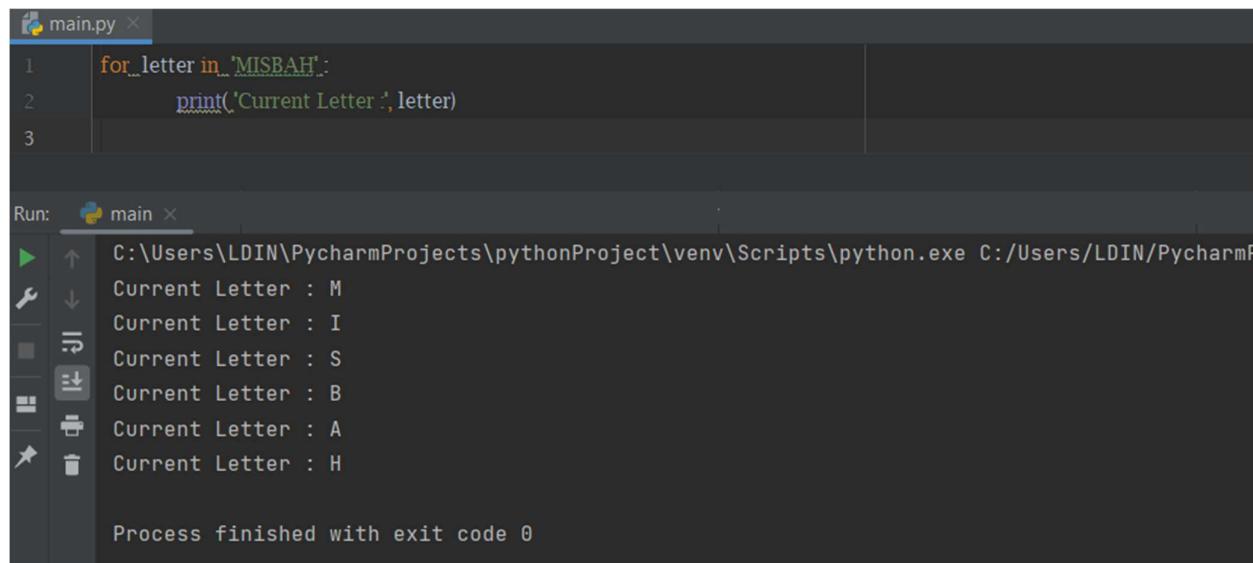
It has the ability to iterate over the items of any sequence, such as a list or a string.

### SYNTAX

for iterating\_var in sequence:

statements(s)

### EXAMPLE:



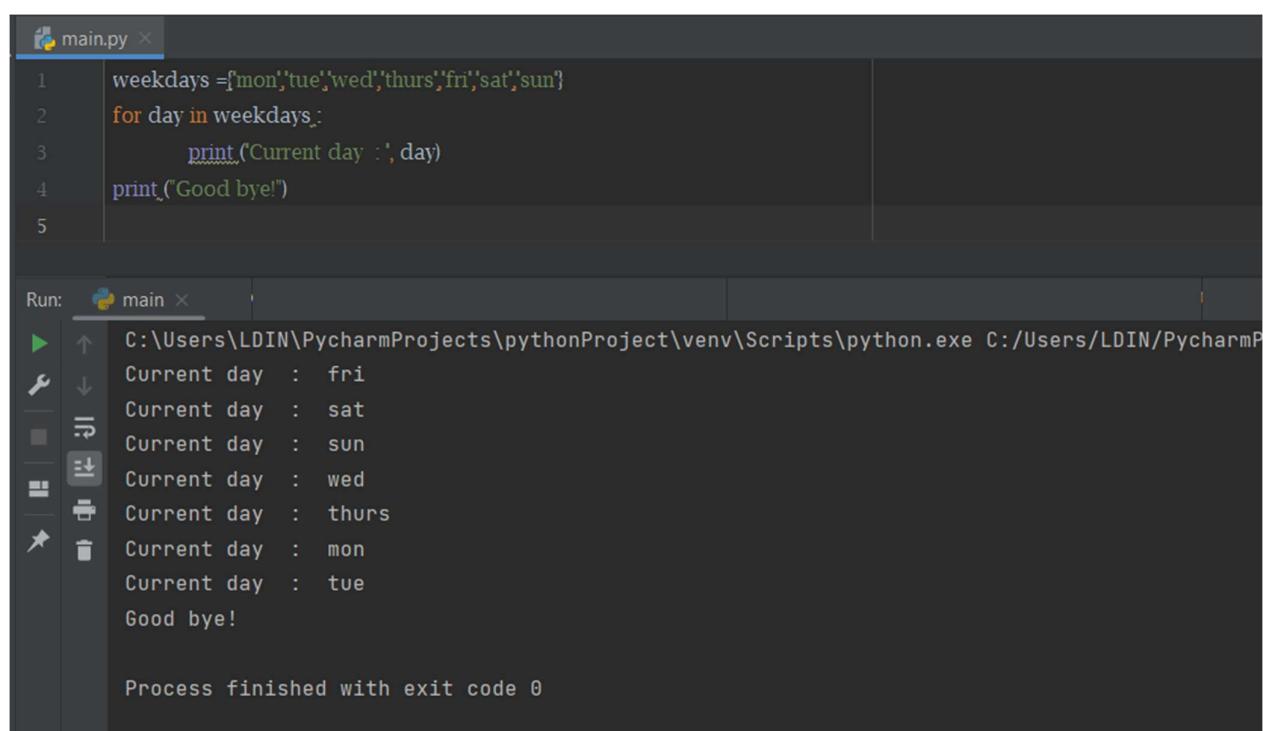
The screenshot shows a PyCharm interface with a code editor and a terminal window. The code editor contains the following Python script:

```
main.py
1 for letter in 'MISBAH':
2     print('Current Letter :', letter)
3 
```

The terminal window shows the output of the script:

```
Run: main ×
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmP
▶ ↑ Current Letter : M
↙ ↓ Current Letter : I
Current Letter : S
Current Letter : B
Current Letter : A
Current Letter : H

Process finished with exit code 0
```



The screenshot shows a PyCharm interface with a code editor and a terminal window. The code editor contains the following Python script:

```
main.py
1 weekdays =['mon','tue','wed','thurs','fri','sat','sun']
2 for day in weekdays:
3     print('Current day :', day)
4 print("Good bye!")
5 
```

The terminal window shows the output of the script:

```
Run: main ×
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmP
▶ ↑ Current day : fri
↙ ↓ Current day : sat
Current day : sun
Current day : wed
Current day : thurs
Current day : mon
Current day : tue
Good bye!

Process finished with exit code 0
```

---

## C) NESTED LOOPS:

---

Python programming language allows to use one loop inside another loop. Following section shows few examples to illustrate the concept.

### SYNTAX

for iterating\_var in sequence:

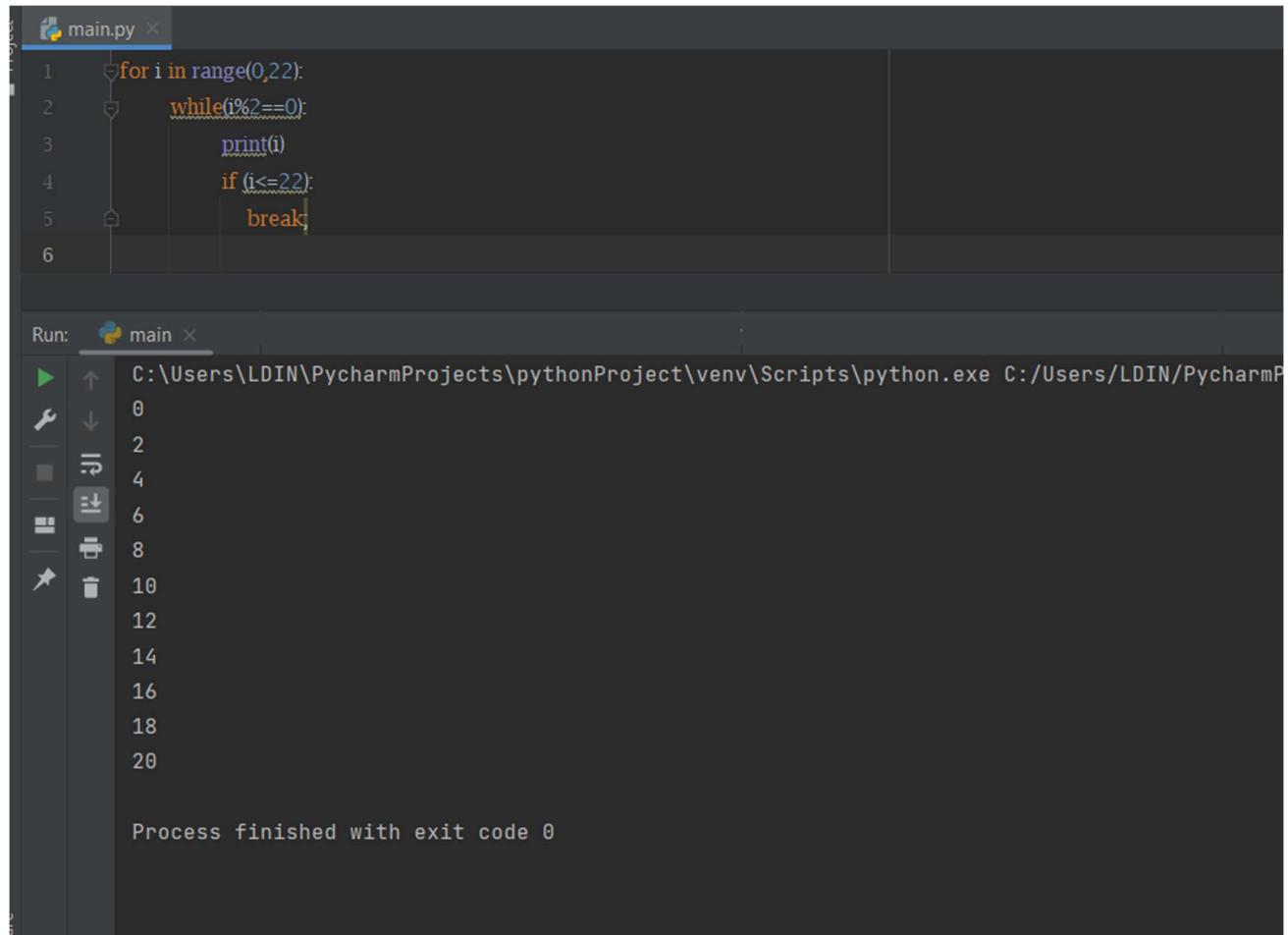
    for iterating\_var in sequence:

        statements(s)

    statements(s)

### EXAMPLE:

The code below uses two nested loops, an if statement and break statement to print all the even numbers from 0 to 20



The screenshot shows the PyCharm IDE interface. The top window displays the Python script 'main.py' with the following code:

```
1  for i in range(0,22):
2      while(i%2==0):
3          print(i)
4          if (i<=22):
5              break;
```

The bottom window shows the 'Run' tab with the output of the script:

```
Run: main x
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmP
0
2
4
6
8
10
12
14
16
18
20

Process finished with exit code 0
```

## LOOP MANIPULATION STATEMENTS

---

### BREAK STATEMENT

---

We use break statement to terminate the current loop. When break statement is encountered inside a loop, control terminates execution of the current loop and control is transferred to the next executable statement

#### SYNTAX: -

break

#### EXAMPLE

```
main.py
1 count=0
2 while count < 10:
3     count += 1
4     if count == 5:
5         break
6     print("inside loop", count)
7 print("out of while loop")
8
```

Run: main

```
C:/Users/LDIN/PycharmProjects/pythonProject/venv/Scripts/python.exe C:/Users/LDIN/PycharmProjects/pythonProject/main.py
inside loop 1
inside loop 2
inside loop 3
inside loop 4
out of while loop

Process finished with exit code 0
```

---

### CONTINUE STATEMENT

---

The continue statement gives you way to skip over the current iteration of any loop. When a continue statement is encountered in the loop, the python interpreter ignores rest of statements in the loop body for current iteration and returns the program execution to the very first statement in the loop body. It does not terminate the loop rather continues with the next iteration.

## **SYNTAX: -**

continue

## **EXAMPLE**

```
main.py
1 for i in range(1,44):
2     if(i%4==0):
3         print(i)
4     else:
5         continue
6

Run: main
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/
4
8
12
16
20
24
28
32
36
40

Process finished with exit code 0
```

---

## **PASS STATEMENT**

---

Pass statement is used to execute noting.it does nothing at all, also called as a null operation

## **SYNTAX: -**

pass

## **EXAMPLE**

The screenshot shows the PyCharm IDE interface. The top window is titled 'main.py' and contains the following Python code:

```
1 for letter in 'misbah':
2     if(letter=='i' or letter=='h'):
3         pass
4     else:
5         print(letter)
6
```

The bottom window is titled 'Run: main' and shows the terminal output of the script execution:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/ldin/PycharmProjects/pythonProject/main.py
m
s
b
a

Process finished with exit code 0
```

---

## XI. EXERCISE

---

1. PRINT THE FOLLOWING PATTERNS USING LOOP:

a. \*

\*\*

\*\*\*

\*\*\*\*

b. \*

\*\*\*

\*\*\*\*\*

\*\*\*

\*

c. 1010101

10101

101

1

```
main.py
1 for i in range(1,5):
2     print('*'*i)
3     print(' '*15)
4     k = 0
5     for i in range(1, 3+1):
6         for space in range(1, (3-i)+1):
7             print(end=' ')
8         while k!=2**i-1:
9             print(' ', end='')
10            k += 1
11        k = 0
12        print()
13    for i in range(3, 1, -1):
14        for space in range(0, 4-i):
15            print(' ', end='')
16        for j in range(i, 2**i-1):
17            print(' ', end='')
18        for j in range(1, i-1):
19            print(' ', end='')
20        print()
21
```

```
main
* 
** 
*** 
**** 
----- 
* 
* * * 
* * * * 
* * * 
* 
```

```
main.py
1 i = 7
2 s = 0
3 k=3
4 while (i>=0 and k>=0):
5     a="{}{}{}{}{}{}{}{}".format(s,i,k,s,i,k,s,i)
6     print(a)
7     k=k-1
8     i=i-2
9     s=s+1
```

```
Run: main
C:/Users/LDIN/PycharmProjects/pythonProject/venv/Scripts/python.exe C:/Users/LDIN/main.py
1010101
10101
101
1
```

2. WRITE A PYTHON PROGRAM TO PRINT ALL EVEN NUMBERS BETWEEN 1 TO 100 USING WHILE LOOP.

```
i = 0
while i < 101:
    if i % 2 == 0:
        print(f'{i}')
    i = i + 1
```

Run: main

60  
62  
64  
66  
68  
70  
72  
74  
76  
78  
80  
82  
84  
86  
88  
90  
92  
94  
96  
98  
100

Process finished with exit code 0

3. WRITE A PYTHON PROGRAM TO FIND THE SUM OF FIRST 10 NATURAL NUMBERS USING FOR LOOP

```
sum=0
for i in range(0,11):
    sum=sum+i
print("Sum of first 10 natural numbers : ",sum)
```

Run: main

C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/...  
Sum of first 10 natural numbers : 55

Process finished with exit code 0

#### 4. WRITE A PYTHON PROGRAM TO PRINT FIBONACCI SERIES.

The screenshot shows the PyCharm IDE interface. The top window is titled 'main.py' and contains Python code to print a Fibonacci series. The bottom window is titled 'Run: main' and shows the execution results.

```
1 nt = int(input("How many terms?"))
2 n1, n2 = 0, 1
3 count = 0
4 if nt <= 0:
5     print("Please enter a positive integer")
6 elif nt == 1:
7     print(f'Fibonacci series upto {nt}:')
8     print(n1)
9 else:
10    print("Fibonacci series:")
11    while count < nt:
12        print(n1)
13        nth = n1 + n2
14        n1 = n2
15        n2 = nth
16        count += 1
```

Run: main

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/main.py
How many terms:9
Fibonacci series:
0
1
1
2
3
5
8
13
21
```

#### 5. WRITE A PYTHON PROGRAM TO CALCULATE FACTORIAL OF A NUMBER

The screenshot shows the PyCharm IDE interface. The top window is titled 'main.py' and contains Python code to calculate the factorial of a number. The bottom window is titled 'Run: main' and shows the execution results.

```
1 n = int(input("Enter a number: "))
2 f = 1
3 if n < 0:
4     print("Factorial does not exist for negative numbers")
5 elif n == 0:
6     print("The factorial of 0 is 1")
7 else:
8     for i in range(1, n + 1):
9         f = f * i
10    print(f'The factorial of {n} is {f}')
11
```

Run: main

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/main.py
Enter a number: 8
The factorial of 8 is 40320
Process finished with exit code 0
```

## 6. WRITE A PYTHON PROGRAM TO REVERSE A GIVEN NUMBER

The screenshot shows the PyCharm IDE interface. The code editor window is titled 'main.py' and contains the following Python code:

```
1 n = int(input("Enter number: "))
2 re = 0
3 while n != 0:
4     d = n % 10
5     re = re * 10 + d
6     n //= 10
7 print("Reversed number: " + str(re))
8
```

The run configuration window is titled 'Run: main'. It shows the command 'C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe' and the output of the program's execution:

```
Enter number: 568173
Reversed number: 371865
Process finished with exit code 0
```

## 7. WRITE A PYTHON PROGRAM TAKES IN A NUMBER AND FINDS THE SUM OF DIGITS IN A NUMBER.

The screenshot shows the PyCharm IDE interface. The code editor window is titled 'main.py' and contains the following Python code:

```
1 n=input("Enter Number:")
2 sum = 0
3 for digit in str(n):
4     sum += int(digit)
5 print(f"Sum of digits in {n}: {sum}")
6
```

The run configuration window is titled 'Run: main'. It shows the command 'C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe' and the output of the program's execution:

```
Enter Number:68974
Sum of digits in 68974: 34
Process finished with exit code 0
```

8. WRITE A PYTHON PROGRAM THAT TAKES A NUMBER AND CHECKS WHETHER IT IS A PALINDROME OR NOT.

The screenshot shows the PyCharm IDE interface. The code editor window is titled "main.py" and contains the following Python script:

```
n=int(input("Enter number:"))
temp=n
rev=0
while n>0:
    dig=n%10
    rev=rev*10+dig
    n=n/10
if temp==rev:
    print("The number is palindrome!")
else:
    print("The number is not palindrome!")

if temp==rev
```

The "Run" tab is selected in the toolbar, and the output window shows the results of running the script with the input "1234321". The output is:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/
Enter number:1234321
The number is palindrome!

Process finished with exit code 0
```

The screenshot shows the PyCharm IDE interface. The code editor window is titled "main.py" and contains the same Python script as the first screenshot.

```
n=int(input("Enter number:"))
temp=n
rev=0
while n>0:
    dig=n%10
    rev=rev*10+dig
    n=n/10
if temp==rev:
    print("The number is palindrome!")
else:
    print("The number is not palindrome!")

if temp==rev
```

The "Run" tab is selected in the toolbar, and the output window shows the results of running the script with the input "4276". The output is:

```
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/
Enter number:4276
The number is not palindrome!

Process finished with exit code 0
```

# PRACTICAL NO. 8:

## WRITE PYTHON PROGRAM TO PERFORM FOLLOWING OPERATIONS ON SET: CREATE SET, ACCESS SET ELEMENTS, UPDATE SET, DELETE SET

### SETS

1. Sets are immutable data structures in Python
2. Values are enclosed in {} curly braces and separated by commas
3. It is an unordered collection of unique items i.e., it doesn't contain duplicate elements
4. When we print a set, it isn't usually printed in the order in which it was declared
5. Sets are unindexed, values cannot be accessed using index position

### CREATING A SET

We can create set in following two ways

1.By using following Syntax

**setName= {val1, val2, val2.... valn}**

2.By using set() function

**setName=set(value(s))**

The screenshot shows a Visual Studio Code interface with a dark theme. The top bar has tabs for Terminal, Help, and a file named 'Practical8\_SET.py'. The main editor area contains the following Python code:

```
1 #CREATING A SET
2 set1={} #EMPTY SET
3 numset={1,4,9,16,25} #INTEGER SET
4 stringset={'red','blue','orange','yellow'}
5 print(set1)
6 print(numset)
7 print(stringset)
8
9 #CREATING A SET WITH SET() FUNCTION
10 s1=set()
11 s2=set(range(1,6))
12 s3=set('abc')
13 print(s1)
14 print(s2)
15 print(s3)
```

Below the editor, there are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing the command line output:

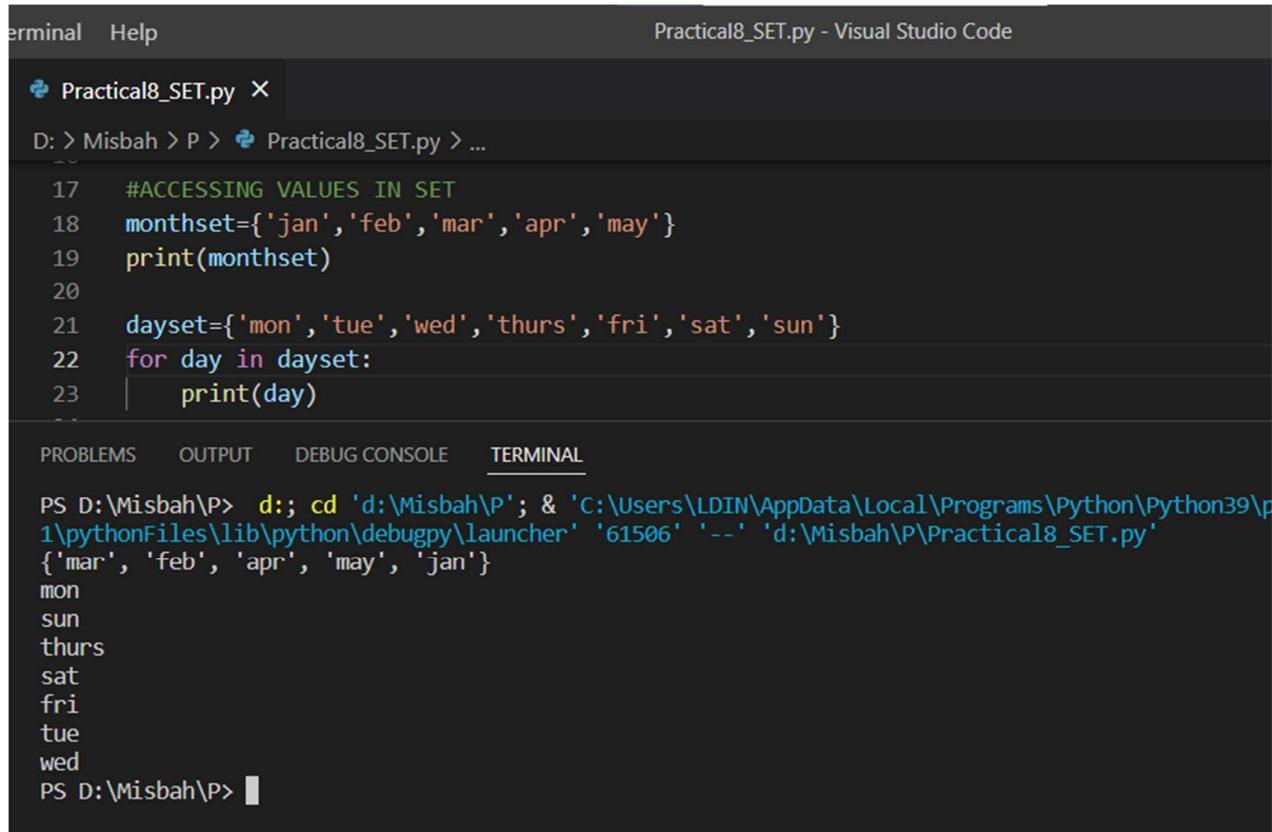
```
PS D:\Misbah\P> d:; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\LDIN\.vscode\1\pythonFiles\lib\python\debugpy\launcher' '61375' '--' 'd:\Misbah\P\Practical8_SET.py'
{}
{16, 1, 4, 9, 25}
{'blue', 'yellow', 'red', 'orange'}
set()
{1, 2, 3, 4, 5}
{'b', 'c', 'a'}
```

---

## ACCESSING VALUES IN A SET

---

We can access values in a set by simply printing the set name or by using a for loop



The screenshot shows a Visual Studio Code interface. The terminal tab is active, displaying Python code and its execution output.

```
terminal  Help
Practical8_SET.py - Visual Studio Code

Practical8_SET.py ×

D: > Misbah > P > Practical8_SET.py > ...

17 #ACCESING VALUES IN SET
18 monthset={'jan', 'feb', 'mar', 'apr', 'may'}
19 print(monthset)
20
21 dayset={'mon', 'tue', 'wed', 'thurs', 'fri', 'sat', 'sun'}
22 for day in dayset:
23     print(day)

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS D:\Misbah\P> d;; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\pythonFiles\lib\python\debugpy\launcher' '61506' '--' 'd:\Misbah\P\Practical8_SET.py'
{'mar', 'feb', 'apr', 'may', 'jan'}
mon
sun
thurs
sat
fri
tue
wed
PS D:\Misbah\P>
```

---

## UPDATING VALUES IN A SET

---

We can update values in Set by using add() function and update() function

### 1. add()

It adds elements to the set and can take only one argument

#### SYNTAX:

setName.add(element)

### 2. update()

With the help of this method, we can add set to a set, list to a set, string to set and tuples to a set, in all cases duplicate elements are not included

#### SYNTAX:

setName.update(setName | listName | tupleName | String)

Practical8\_SET.py X

```
D: > Misbah > P > Practical8_SET.py > ...  
25     #UPDATING VALUES IN SET  
26     #USING ADD() METHOD  
27     fruits={'banana','apple','guava'}  
28     fruits.add('pear')  
29     print(fruits)  
30  
31     set1={10,20,30}  
32     list1=[40,50]  
33     set1.update(list1)  
34     print(set1)  
35  
36     #USING UPDATE() METHOD  
37     info1={'NAME : MISBAH'}  
38     info2={'ROLLNO : 15'}  
39     info1.update(info2)  
40     print(info1)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\Misbah\P> d;; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\3.9\pythonFiles\lib\python\debugpy\launcher' '58935' '--' 'd:\Misbah\P\Practical8_SET.py'  
{'apple', 'guava', 'pear', 'banana'}  
{40, 10, 50, 20, 30}  
{'ROLLNO : 15', 'NAME : MISBAH'}  
PS D:\Misbah\P>
```

---

## DELETING VALUES IN A SET

---

Deletion of values in set is possible with the help of the following four methods

- `pop()` : removes random item or last item from the set
- `discard()` : deletes the item specified
- `remove()`: deletes the item specified ,except it raises a `KeyError` if element doesn't exist
- `clear()` : removes all elements from the set

**Practical8\_SET.py X**

```
D: > Misbah > P > Practical8_SET.py > ...
41
42     #DELETING VALUES IN A SET
43
44     flowers={'rose','jasmine','papaya','lily','cherry'}
45     print(flowers)
46     flowers.discard('papaya')
47     print(flowers)
48     flowers.remove('jasmine')
49     print(flowers)
50     flowers.pop()
51     print(flowers)
52     flowers.clear()
53     print(flowers)
54
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\Misbah\P> d:; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python31\pythonFiles\lib\python\debugpy\launcher' '59019' '--' 'd:\Misbah\P\Practical8_SET.py'
{'papaya', 'cherry', 'lily', 'rose', 'jasmine'}
{'cherry', 'lily', 'rose', 'jasmine'}
{'cherry', 'lily', 'rose'}
{'lily', 'rose'}
set()
PS D:\Misbah\P> []
```

---

**BASIC SET OPERATIONS**

---

Sets can be used to carry out mathematical set operations like *union*, *intersection*, *difference* and *symmetric difference*. We can do this with operators or methods. Consider following two sets

$$A = \{1, 2, 3, 4, 5\}$$

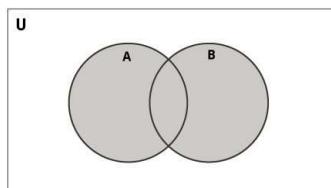
$$B = \{4, 5, 6, 7, 8\}$$

**1.UNION**

Union of A and B is a set of all elements from both sets.

Union is performed using | operator.

Same can be accomplished using the union() method.

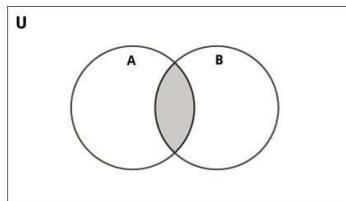


## 2.SET INTERSECTION

Intersection of A and B is a set of elements that are common in both the sets.

Intersection is performed using & operator.

Same can be accomplished using the intersection() method.

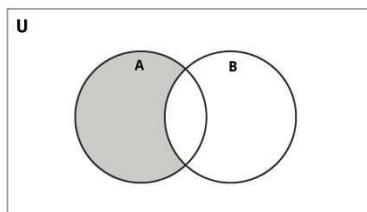


## 3.SET DIFFERENCE

Difference of the set B from set A ( $A - B$ ) is a set of elements that are only in A but not in B. Similarly,  $B - A$  is a set of elements in B but not in A.

Difference is performed using - operator.

Same can be accomplished using the difference() method.

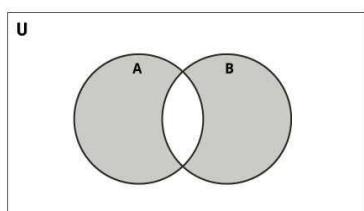


## 4.SET SYMMETRIC DIFFERENCE

Symmetric Difference of A and B is a set of elements in A and B but not in both (excluding the intersection).

Symmetric difference is performed using ^ operator.

Same can be accomplished using the method symmetric\_difference()



## SET OPERATIONS USING OPERATORS

```
D: > Misbah > P > Practical8_SET.py > ...
56 #SET OPERATIONS
57
58 A = {1, 2, 3, 4, 5}
59 B = {4, 5, 6, 7, 8}
60 print("Set union method")
61 print(A | B)
62 print("Set intersection method")
63 print(A & B)
64 print("Set difference method")
65 print(A - B)
66 print("Set symmetric difference method")
67 print(A ^ B)
68

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS D:\Misbah\Practical8_SET.py> d:; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\LDIN\OneDrive\Desktop\pythonFiles\lib\python\debugpy\launcher' '65362' '--' 'd:\Misbah\P\Practical8_SET.py'
Set union method
{1, 2, 3, 4, 5, 6, 7, 8}
Set intersection method
{4, 5}
Set difference method
{1, 2, 3}
Set symmetric difference method
{1, 2, 3, 6, 7, 8}
PS D:\Misbah\Practical8_SET.py>
```

## SET OPERATIONS USING INBUILT FUNCTIONS

```
D: > Misbah > P > Practical8_SET.py > ...
67 A = {1, 2, 3, 4, 5}
68 B = {4, 5, 6, 7, 8}
69 #WITH FUNCTIONS
70 print("Set union method")
71 print(A.union(B))
72 print(B.union(A))
73 print("Set intersection method")
74 print(A.intersection(B))
75 print(B.intersection(A))
76 print("Set difference method")
77 print(A.difference(B))
78 print(B.difference(A))
79 print("Set symmetric difference method")
80 print(A.symmetric_difference(B))
81 print(B.symmetric_difference(A))
82

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS D:\Misbah\Practical8_SET.py> & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\LDIN\OneDrive\Desktop\pythonFiles\lib\python\debugpy\launcher' '65435' '--' 'd:\Misbah\P\Practical8_SET.py'
Set union method
{1, 2, 3, 4, 5, 6, 7, 8}
{1, 2, 3, 4, 5, 6, 7, 8}
Set intersection method
{4, 5}
{4, 5}
Set difference method
{1, 2, 3}
{8, 6, 7}
Set symmetric difference method
{1, 2, 3, 6, 7, 8}
{1, 2, 3, 6, 7, 8}
PS D:\Misbah\Practical8_SET.py>
```

## XI. EXERCISE

1. WRITE A PYTHON PROGRAM TO CREATE A SET, ADD MEMBER(S) IN A SET AND REMOVE ONE ITEM FROM SET.

```
D: > Misbah > P > Practical8_SET.py > ...
82
83     s1={12,45,32,76,14}
84     print(f'Elements in {s1} before updating')
85     s1.add(67)
86     s1.add(98)
87     print(f'Elements in {s1} after updating')
88     s1.pop()
89     print(f'Elements in {s1} after using pop')
90     s1.discard(76)
91     print(f'Elements in {s1} after using discard')
92     s1.remove(98)
93     print(f'Elements in {s1} after using remove')
94

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Misbah\Practical8_SET.py> & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\LDIN\.vscode\on/debugpy\launcher' '65458' '--' 'd:\Misbah\Practical8_SET.py'
Elements in {32, 12, 45, 14, 76} before updating
Elements in {32, 98, 67, 12, 45, 14, 76} after updating
Elements in {98, 67, 12, 45, 14, 76} after using pop
Elements in {98, 67, 12, 45, 14} after using discard
Elements in {67, 12, 45, 14} after using remove
PS D:\Misbah\Practical8_SET.py>
```

2. WRITE A PYTHON PROGRAM TO PERFORM FOLLOWING OPERATIONS ON SET: INTERSECTION OF SETS, UNION OF SETS, SET DIFFERENCE, SYMMETRIC DIFFERENCE, CLEAR A SET.

```
Practical8_SET.py X

D: > Misbah > P > Practical8_SET.py > ...
83     A = {0, 2, 4, 6, 8}
84     B = {1, 2, 3, 4, 5}
85     print("Union :", A | B)
86     print("Intersection :", A & B)
87     print("Difference :", A - B)
88     print("Symmetric difference :", A ^ B)
89

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Misbah\Practical8_SET.py> & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\LDIN\.vscode\on/debugpy\launcher' '65494' '--' 'd:\Misbah\Practical8_SET.py'
Union : {0, 1, 2, 3, 4, 5, 6, 8}
Intersection : {2, 4}
Difference : {0, 8, 6}
Symmetric difference : {0, 1, 3, 5, 6, 8}
PS D:\Misbah\Practical8_SET.py> []
```

3. WRITE A PYTHON PROGRAM TO FIND MAXIMUM AND THE MINIMUM VALUE IN A SET.

```
D: > Misbah > P > Practical8_SET.py > ...
83
84     s={1,4,6,7,89,56,3,62,66,30,2}
85     print(f'Largest number from Set {s}:',max(s))
86     print(f'Smallest number from Set {s}:',min(s))
87

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Misbah\P> & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\on\debugpy\launcher' '49177' '--' 'd:\Misbah\P\Practical8_SET.py'
Largest number from Set {1, 2, 66, 3, 4, 6, 7, 62, 56, 89, 30}: 89
Smallest number from Set {1, 2, 66, 3, 4, 6, 7, 62, 56, 89, 30}: 1
PS D:\Misbah\P>
```

4. WRITE A PYTHON PROGRAM TO FIND THE LENGTH OF A SET.

```
Practical8_SET.py ●

D: > Misbah > P > Practical8_SET.py > ...
82
83     # 4.      WRITE A PYTHON PROGRAM TO FIND THE LENGTH OF A SET.
84
85     s={1,4,6,7,89,56,3,62,66,30,2}
86     print(f'Length of Set {s}:',len(s))
87

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS D:\Misbah\P> d;; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\Py
1\pythonFiles\lib\python\debugpy\launcher' '49187' '--' 'd:\Misbah\P\Practical8_SET.py'
Length of Set {1, 2, 66, 3, 4, 6, 7, 62, 56, 89, 30}: 11
PS D:\Misbah\P>
```

# PRACTICAL NO. 9:

## WRITE PYTHON PROGRAM TO PERFORM FOLLOWING OPERATIONS ON DICTIONARIES: CREATE DICTIONARY, ACCESS DICTIONARY ELEMENTS, UPDATE DICTIONARY, DELETE DICTIONARY, LOOPING THROUGH DICTIONARY

Dictionaries are associative data structures in Python

- It is not always possible to remember the index value of each element
- Hence, Dictionaries help us to identify an element with the help of its key
- Dictionaries contain key: values pair and each pair are separated by commas

---

### CREATING A DICTIONARY

---

- We can create a dictionary by enclosing *key: value* pairs in curly braces

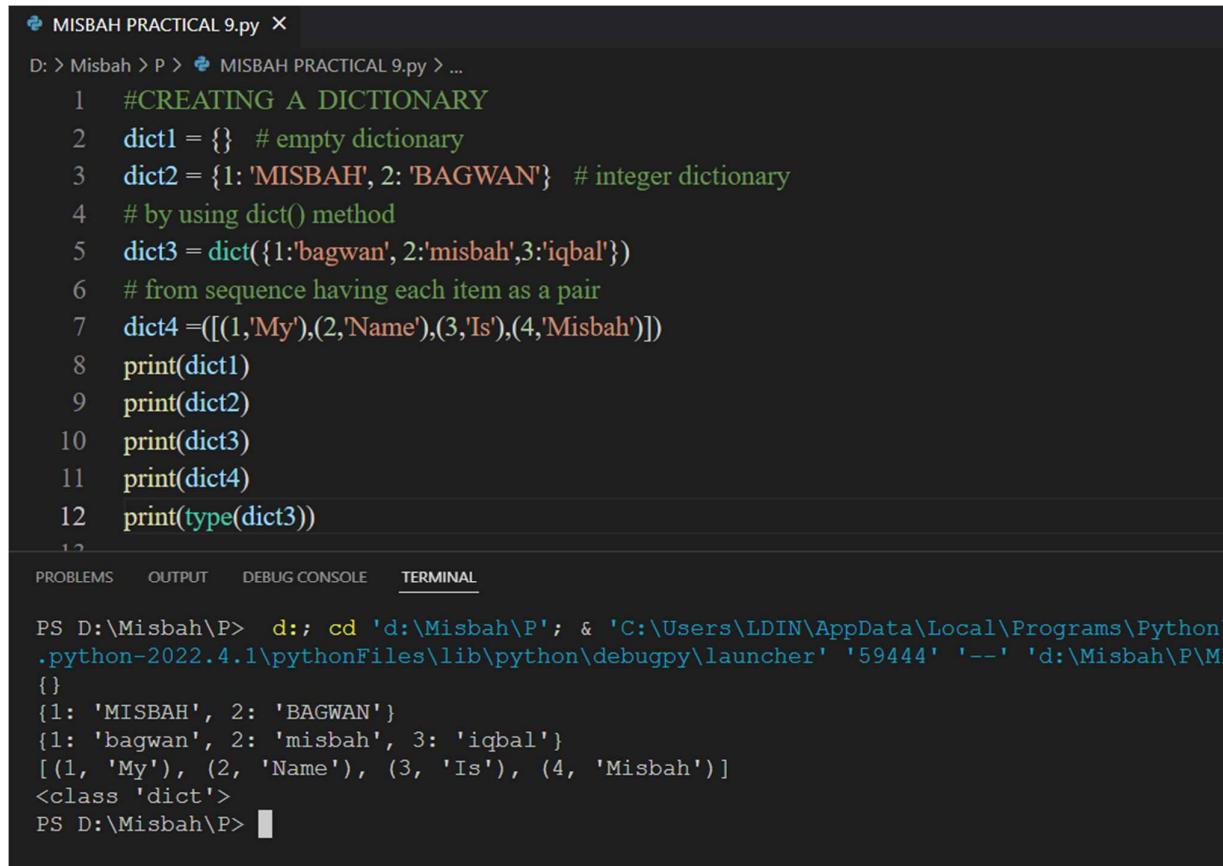
#### SYNTAX

```
dict1= {key1: val1, key2: val2, key3: val3,....., keyn: valn}
```

- We can directly use dict() function

#### SYNTAX

```
dict1= ({key1: val1, key2: val2, key3: val3,....., keyn: valn})
```



The screenshot shows a code editor window with a dark theme. The file is named 'MISBAH PRACTICAL 9.py'. The code is as follows:

```
 MISBAH PRACTICAL 9.py ×
D: > Misbah > P > MISBAH PRACTICAL 9.py > ...
1 #CREATING A DICTIONARY
2 dict1 = {} # empty dictionary
3 dict2 = {1: 'MISBAH', 2: 'BAGWAN'} # integer dictionary
4 # by using dict() method
5 dict3 = dict({1:'bagwan', 2:'misbah',3:'iqbal'})
6 # from sequence having each item as a pair
7 dict4=[(1,'My'),(2,'Name'),(3,'Is'),(4,'Misbah')]
8 print(dict1)
9 print(dict2)
10 print(dict3)
11 print(dict4)
12 print(type(dict3))
13
```

Below the code, there are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing the command line output:

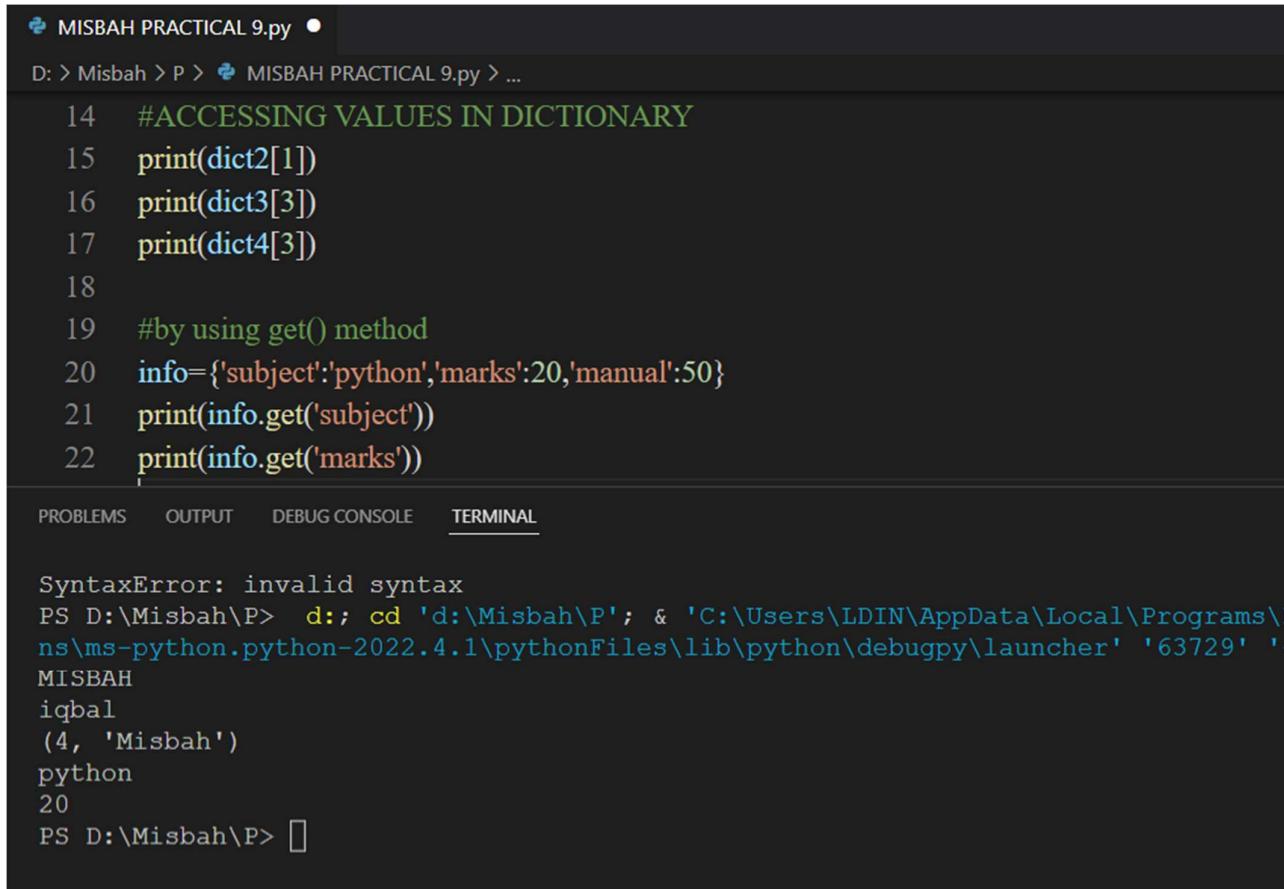
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\Misbah\P> d;; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '59444' '--' 'd:\Misbah\P\MISBAH PRACTICAL 9.py'
{}
{1: 'MISBAH', 2: 'BAGWAN'}
{1: 'bagwan', 2: 'misbah', 3: 'iqbal'}
[(1, 'My'), (2, 'Name'), (3, 'Is'), (4, 'Misbah')]
<class 'dict'>
PS D:\Misbah\P>
```

---

## ACCESSING VALUES IN DICTIONARY

---

- We can refer to its key name, inside square brackets[]
- We can also use get() method to access value of a key



```
MISBAH PRACTICAL 9.py
D: > Misbah > P > MISBAH PRACTICAL 9.py > ...
14 #ACCESSING VALUES IN DICTIONARY
15 print(dict2[1])
16 print(dict3[3])
17 print(dict4[3])
18
19 #by using get() method
20 info={'subject':'python','marks':20,'manual':50}
21 print(info.get('subject'))
22 print(info.get('marks'))
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

```
SyntaxError: invalid syntax
PS D:\Misbah\P> d;; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\Python\ms-python.python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '63729' 'MISBAH
iqbal
(4, 'Misbah')
python
20
PS D:\Misbah\P> [ ]
```

---

## UPDATING DICTIONARY

---

- Dictionaries are mutable datatypes meaning they can be modified accordingly
- We can add new items or change the value of existing items using an **assignment operator**.
- If the key is already present, then the existing value gets updated. In case the key is not present, a new (key: value) pair is added to the dictionary.

---

## SYNTAX

```
dict1[existing_key] = diff_value
dict1[new_key]=new_key
```

## MISBAH PRACTICAL 9.py X

D: > Misbah > P > MISBAH PRACTICAL 9.py > ...

```
24 #UPDATING VALUES IN DICTIONARY
25 mydict = {'name': 'MISBAH', 'age': 17}
26 mydict['age'] = 18
27 print(mydict)
28 mydict['address'] = 'Dhankawadi'
29 print(mydict)
30
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

```
PS D:\Misbah\P> & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python
nFiles\lib\python\debugpy\launcher' '63771' '--' 'd:\Misbah\P\MISBAH PRACTICAL
{'name': 'MISBAH', 'age': 18}
{'name': 'MISBAH', 'age': 18, 'address': 'Dhankawadi'}
PS D:\Misbah\P>
```

---

## REMOVING ELEMENTS FROM DICTIONARY

---

With the help of the following methods, it is easy to remove an item from the dictionary by specifying its key, deleting the last key: value pair, emptying the entire dictionary by removing all elements (clearing it) and deleting the entire dictionary

- `pop(key)`: removes specified key: value pair from the dictionary
- `popitem()`: removes last element from the dictionary
- `clear()`; removes all pairs from the dictionary
- `del keyword`: deletes the entire dictionary

## MISBAH PRACTICAL 9.py

```
D: > Misbah > P > MISBAH PRACTICAL 9.py > ...
31 # Removing elements from a dictionary
32 squares = {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
33 print(squares.pop(4))
34 print(squares)
35 print(squares.popitem())
36 print(squares)
37 squares.clear()
38 print(squares)
39
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
```

```
Try the new cross-platform PowerShell https://aka.ms/pscore6
```

```
PS D:\Misbah\P> & 'C:\Users\LDIN\AppData\Local\Programs\Python\Python39\python.exe' 'nFiles\lib\python\debugpy\launcher' '55096' '--' 'd:\Misbah\P\MISBAH PRACTICAL 9.py'
16
{1: 1, 2: 4, 3: 9, 5: 25}
(5, 25)
{1: 1, 2: 4, 3: 9}
{}
PS D:\Misbah\P>
```

## BUILT IN FUNCTIONS AND METHODS IN DICTIONARY

Function	Description
all()	Return True if all keys of the dictionary are True (or if the dictionary is empty).
any()	Return True if any key of the dictionary is true. If the dictionary is empty, return False.
len()	Return the length (the number of items) in the dictionary.
Sorted()	Return a new sorted list of keys in the dictionary.

---

## EXAMPLE

---

```
 MISBAH PRACTICAL 9.py ●
D: > Misbah > P > MISBAH PRACTICAL 9.py > ...
30
31 # Dictionary Built-in Functions
32 squares = {0: 0, 1: 1, 3: 9, 5: 25, 7: 49, 9: 81}
33 print(all(squares))
34 print(any(squares))
35 print(len(squares))
36 print(sorted(squares))
37

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS D:\Misbah\P> d;; cd 'd:\Misbah\P'; & 'C:\Users\LDIN\AppData\Local\Programs\python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '63421' '--' 'd:\Misb
False
True
6
[0, 1, 3, 5, 7, 9]
PS D:\Misbah\P>
```

---

## LOOPING THROUGH DICTIONARY

---

A dictionary can be iterated using the for loop.

```
 MISBAH PRACTICAL 9.py ●
D: > Misbah > P > MISBAH PRACTICAL 9.py > ...
38 month={1:'sun',2:'mon',3:'tue',4:'wed',5:'thurs',6:'fri',7:'sat'}
39 for key,value in month.items():
40     print(key,'-',value)
41

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\Users\LDIN> & C:/Users/LDIN/AppData/Local/Programs/Python/Python39/python
1 - sun
2 - mon
3 - tue
4 - wed
5 - thurs
6 - fri
7 - sat
PS C:\Users\LDIN>
```

## XI. EXERCISE

1. WRITE A PYTHON SCRIPT TO SORT (ASCENDING AND DESCENDING) A DICTIONARY BY VALUE.

EXP9A.py - C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9A.py (3.9.10)

```
File Edit Format Run Options Window Help
d={"One":1, "Three": 3, "Zero": 0, "Five": 5, "Four":4}
print(fDictionary before sorting:{d})
li=list(d.items())
l=len(li)
for i in range(l-1):
    for j in range(i+1,l):
        if li[i][1]>li[j][1]:
            t=li[i]
            li[i]=li[j]
            li[j]=t
sortd=dict(li)
print(fDictionary after sorting by value:{sortd})
```

IDLE Shell 3.9.10

File Edit Shell Debug Options Window Help

Python 3.9.10 (tags/v3.9.10:f2f3f53, Jan 17 2022, 15:14:21) [MSC v.1929 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.

>>>

===== RESTART: C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9A.py =====

Dictionary before sorting:{'One': 1, 'Three': 3, 'Zero': 0, 'Five': 5, 'Four': 4}

Dictionary after sorting by value:{'Zero': 0, 'One': 1, 'Three': 3, 'Four': 4, 'Five': 5}

>>> |

2. WRITE A PYTHON SCRIPT TO CONCATENATE FOLLOWING DICTIONARIES TO CREATE A NEW ONE.

a. Sample Dictionary: b. dic1 = {1:10, 2:20} c. dic2 = {3:30, 4:40} d. dic3 = {5:50,6:60}

EXP9B.py - C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9B.py (3.9.10)

File Edit Format Run Options Window Help

```
d1={1:10, 2:20}
d2={3:30, 4:40}
d3={5:50,6:60}
d4 = {}
for i in (d1, d2, d3):
    d4.update(i)
print(d4)
```

IDLE Shell 3.9.10

File Edit Shell Debug Options Window Help

Python 3.9.10 (tags/v3.9.10:f2f3f53, Jan 17 2022, 15:14:21) [MSC v.1929 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.

>>>

===== RESTART: C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9B.py =====

{1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}

>>> |

### 3. WRITE A PYTHON PROGRAM TO COMBINE TWO DICTIONARY ADDING VALUES FOR COMMON KEYS.

- a. D1 = {'A': 100, 'B': 200, 'C': 300}
- b. D2 = {'A': 300, 'B': 200, 'D': 400}

EXP9C.py - C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9C.py (3.9.10)

File Edit Format Run Options Window Help

```
d1 = {'a': 100, 'b': 200, 'c': 300}
d2 = {'a': 300, 'b': 200, 'd': 400}
for key in d2:
    if key in d1:
        d2[key] = d2[key] + d1[key]
    else:
        pass
res = d1 | d2
print(res)
```

IDLE Shell 3.9.10

File Edit Shell Debug Options Window Help

Python 3.9.10 (tags/v3.9.10:f2f3f53, Jan 17 2022, 15:14:21) [MSC v.1929 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.

>>>

```
===== RESTART: C:/Users/LDIN/AppData/Local/Programs/Python
{'a': 400, 'b': 400, 'c': 300, 'd': 400}
>>> |
```

### 4. WRITE A PYTHON PROGRAM TO PRINT ALL UNIQUE VALUES IN A DICTIONARY.

A. SAMPLE DATA: [{"V": "S001"}, {"V": "S002"}, {"VI": "S001"}, {"VI": "S005"}, {"VII": "S005"}, {"V": "S009"}, {"VIII": "S007"}]

EXP9D.py - C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9D.py (3.9.10)

File Edit Format Run Options Window Help

```
L = [{"V": "S001"}, {"V": "S002"}, {"VI": "S001"}, {"VI": "S005"}, {"VII": "S005"}, {"V": "S009"}, {"VIII": "S007"}]
print("Original List: ", L)
u_value = set(val for dic in L for val in dic.values())
print("Unique Values: ", u_value)
```

IDLE Shell 3.9.10

File Edit Shell Debug Options Window Help

Python 3.9.10 (tags/v3.9.10:f2f3f53, Jan 17 2022, 15:14:21) [MSC v.1929 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.

>>>

```
===== RESTART: C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9D.py =====
Original List: [{"V": "S001"}, {"V": "S002"}, {"VI": "S001"}, {"VI": "S005"}, {"VII": "S005"}, {"V": "S009"}, {"VIII": "S007"}]
Unique Values: {'S009', 'S007', 'S001', 'S002', 'S005'}
>>> |
```

#### 4. WRITE A PYTHON PROGRAM TO FIND THE HIGHEST 3 VALUES IN A DICTIONARY.

```
EXP9E.py - C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9E.py (3.9.10)
File Edit Format Run Options Window Help
d = {"A": 105, "B": 274, "C": 13, "D": 41, "E": 152}
l = list(d.values())
lr = l[0]
se = l[0]
tr = l[0]
for i in l:
    if lr <= i:
        lr = i
    elif se < i:
        if lr != i:
            se = i
    elif tr < i:
        if se != i:
            tr = i
print(f"Highest three values in {d} are {lr},{se},{tr}")
```

```
IDLE Shell 3.9.10
File Edit Shell Debug Options Window Help
Python 3.9.10 (tags/v3.9.10:f2f3f53, Jan 17 2022, 15:14:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/LDIN/AppData/Local/Programs/Python/Python39/EXP9E.py =====
Highest three values in {'A': 105, 'B': 274, 'C': 13, 'D': 41, 'E': 152} are 274,152,105
>>> |
```

## PRACTICAL NO. 10:

- WRITE PYTHON PROGRAM TO DEMONSTRATE MATH BUILT-IN FUNCTIONS (ANY 2 PROGRAMS)
- WRITE PYTHON PROGRAM TO DEMONSTRATE STRING BUILT-IN FUNCTIONS (ANY 2 PROGRAMS)

### PYTHON MATH FUNCTIONS

TO USE THE MATH MODULE IN YOUR PYTHON APPLICATION, FIRST, YOU NEED TO IMPORT THE MODULE USING **IMPORT MATH** STATEMENTS IN YOUR PROGRAM.

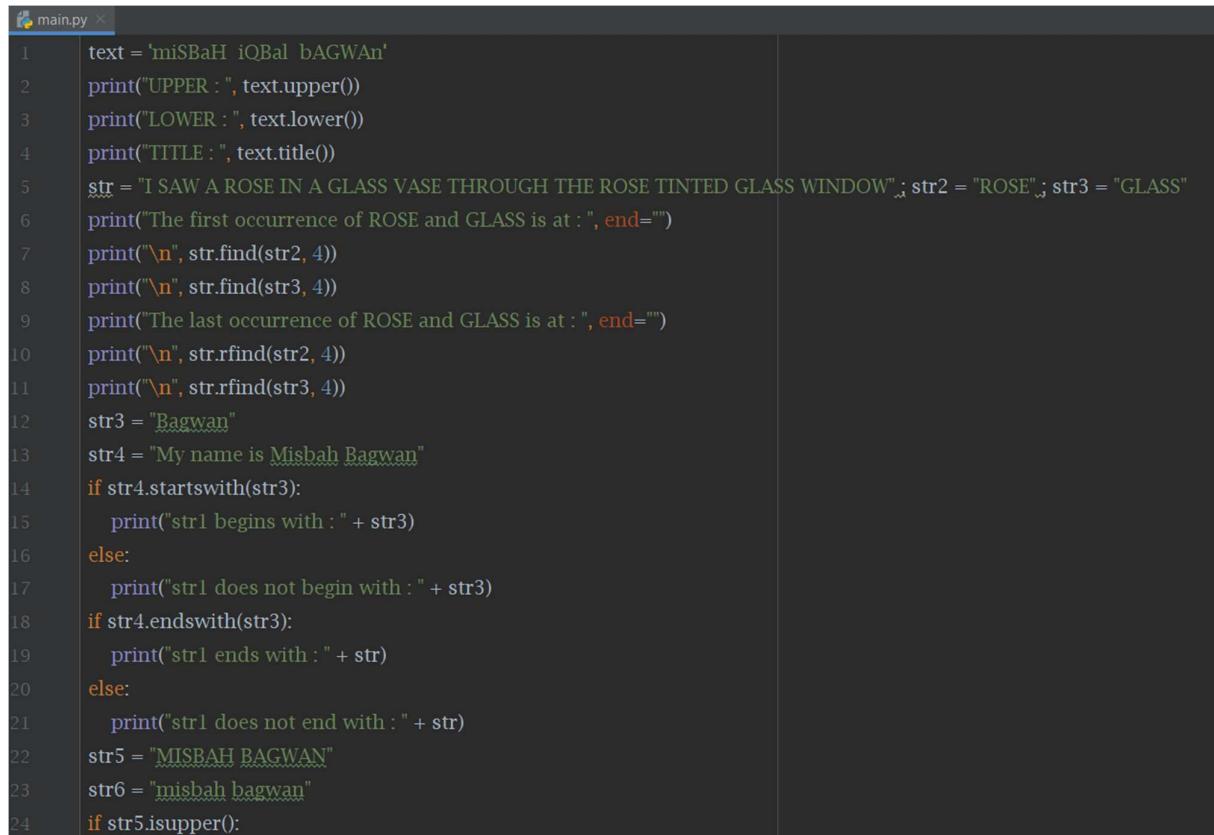
```
main.py
1 import math
2 data = 21.6
3 print("The floor of 21.6 is:", math.floor(data))
4 print(math.ceil(21.6))
5 print('The value of PI:',math.pi)
6 print(math.pow(4, 2))
7 number = -21.19
8 print("The given number is :", number)
9 print('Floor value is :',math.floor(number))
10 print('Ceiling value is :',math.ceil(number))
11 print('Absolute value is :',math.fabs(number))
12 number = 1e-4
13 print("The given number (x) is :", number)
14 print('e^x (using exp() function) is :',math.exp(number)-1)
15 print('log(fabs(x), base) is :',math.log(math.fabs(number), 10))
16 angleInDegree = 90
17 angleInRadian = math.radians(angleInDegree)
18 print("The given angle is :", angleInRadian)
19 print('sin(x) is :',math.sin(angleInRadian))
20 print('cos(x) is :',math.cos(angleInRadian))
21 print('tan(x) is :',math.tan(angleInRadian))
```

```
Run: main
C:/Users/LDIN/PycharmProjects/pythonProject/venv/Scripts/python.exe C:/Users/LDIN/PycharmProjects/
The floor of 21.6 is: 21
22
The value of PI: 3.141592653589793
16.0
The given number is : -21.19
Floor value is : -21
Ceiling value is : -21
Absolute value is : 21.19
The given number (x) is : 0.0001
e^x (using exp() function) is : 0.0001000050001667141
log(fabs(x), base) is : -3.999999999999999
The given angle is : 1.5707963267948966
sin(x) is : 1.0
cos(x) is : 6.123233995736766e-17
tan(x) is : 1.633123935319537e+16

Process finished with exit code 0
```

## PYTHON STRING FUNCTIONS

Method	Description
<code>center()</code>	Returns a centred string
<code>endswith()</code>	Returns true if the string ends with the specified value
<code>find()</code>	Searches the string for a specified value and returns the position of where it was found
<code>isalpha()</code>	Returns True if all characters in the string are in the alphabet
<code>islower()</code>	Returns True if all characters in the string are lower case
<code>isspace()</code>	Returns True if all characters in the string are whitespaces
<code>isupper()</code>	Returns True if all characters in the string are upper case
<code>ljust()</code>	Returns a left justified version of the string
<code>lower()</code>	Converts a string into lower case
<code>replace()</code>	Returns a string where a specified value is replaced with a specified value
<code>rfind()</code>	Searches the string for a specified value and returns the last position of where it was found
<code>rjust()</code>	Returns a right justified version of the string
<code>startswith()</code>	Returns true if the string starts with the specified value
<code>title()</code>	Converts the first character of each word to upper case
<code>upper()</code>	Converts a string into upper case



```
main.py ×
1  text = 'miSBaH iQBal bAGWAn'
2  print("UPPER : ", text.upper())
3  print("LOWER : ", text.lower())
4  print("TITLE : ", text.title())
5  str = "I SAW A ROSE IN A GLASS VASE THROUGH THE ROSE TINTED GLASS WINDOW"; str2 = "ROSE"; str3 = "GLASS"
6  print("The first occurrence of ROSE and GLASS is at : ", end="")
7  print("\n", str.find(str2, 4))
8  print("\n", str.find(str3, 4))
9  print("The last occurrence of ROSE and GLASS is at : ", end="")
10 print("\n", str.rfind(str2, 4))
11 print("\n", str.rfind(str3, 4))
12 str3 = "Bagwan"
13 str4 = "My name is Misbah Bagwan"
14 if str4.startswith(str3):
15     print("str1 begins with : " + str3)
16 else:
17     print("str1 does not begin with : " + str3)
18 if str4.endswith(str3):
19     print("str1 ends with : " + str)
20 else:
21     print("str1 does not end with : " + str)
22 str5 = "MISBAH BAGWAN"
23 str6 = "misbah bagwan"
24 if str5.isupper():
```

```
main.py x
25     print("All characters in str are upper cased")
26 else:
27     print("All characters in str are not upper cased")
28 if str6.islower():
29     print("All characters in str1 are lower cased")
30 else:
31     print("All characters in str1 are not lower cased")
32 str7 = "My name is Misbah Bagwan"
33 print(" The length of string is : ", len(str7))
34 str8 = "Python is an easy programming language"
35 print("The string after centering with '-' is : ", end="")
36 print(str8.center(20, '-'))
37 print("The string after ljust is : ", end="")
38 print(str8.ljust(20, '-'))
39 print("The string after rjust is : ", end="")
40 print(str8.rjust(20, '-'))
41 str9 = "PYTHON MAD ETI "
42 str10 = " 22616"
43 print(str9.isalpha(), str10.isalnum(), str10.isspace())
44 str11 = "i took the elevator and entered the flat"
45 str12 = "lift"
46 str13 = "elevator"
47 print("The string after replacing strings is : ", end="")
48 print(str11.replace(str12, str13, 2))
```

```
main x
C:\Users\LDIN\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/LDIN/PycharmProjects/pythonProject/main.py
UPPER : MISBAH IQBAL BAGWAN
LOWER : misbah iqbal bagwan
TITLE : Misbah Iqbal Bagwan
The first occurrence of ROSE and GLASS is at :
8

18
The last occurrence of ROSE and GLASS is at :
41

53
str1 does not begin with : Bagwan
str1 ends with : I SAW A ROSE IN A GLASS VASE THROUGH THE ROSE TINTED GLASS WINDOW
All characters in str are upper cased
All characters in str1 are lower cased
The length of string is : 24
The string after centering with '-' is : Python is an easy programming language
The string after ljust is : Python is an easy programming language
The string after rjust is : Python is an easy programming language
False False False
The string after replacing strings is : i took the elevator and entered the flat

Process finished with exit code 0
```

## XI. EXERCISE

1. WRITE A PYTHON FUNCTION THAT ACCEPTS A STRING AND CALCULATE THE NUMBER OF UPPER-CASE LETTERS AND LOWER-CASE LETTERS.

The screenshot shows a code editor interface with two tabs: "MISBAH PRACTICAL 9.py" and "PRACTICAL 10.py". The "PRACTICAL 10.py" tab is active, displaying the following Python code:

```
1  s=input('Enter a string: ')
2  def sr(s):
3      il=0
4      iu=0
5      for i in s:
6          if(i.islower()):
7              il=il+1
8          elif(i.isupper()):
9              iu=iu+1
10     print(f'The number of lowercase characters in {s}: {il}')
11     print(f'The number of uppercase characters in {s}: {iu}')
12 sr(s)
```

Below the code, there are four tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab shows the following command-line interaction:

```
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\LDIN> & C:/Users/LDIN/AppData/Local/Programs/Python/Python39/python.exe "d:/Misbah/P/PRACTICAL 10.py"
Enter a string:mIsbAH IqbAL BagWAN
The number of lowercase characters in mIsbAH IqbAL BagWAN: 7
The number of uppercase characters in mIsbAH IqbAL BagWAN: 10
PS C:\Users\LDIN> 
```

2. WRITE A PYTHON PROGRAM TO GENERATE A RANDOM FLOAT WHERE THE VALUE IS BETWEEN 5 AND 50 USING PYTHON MATH MODULE.

The screenshot shows a code editor interface with two tabs: "MISBAH PRACTICAL 9.py" and "PRACTICAL 10.py". The "PRACTICAL 10.py" tab is active, displaying the following Python code:

```
13
14  import random
15  print("Random Float number between 5 to 50:",random.uniform(5.0, 50.0))
16
17 
```

Below the code, there are four tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab shows the following command-line interaction:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Misbah\P> & 'C:/Users/LDIN/AppData/Local/Programs/Python/Python39/python.exe' 'nFiles\lib\python\debugpy\launcher' '60228' '--' 'd:\Misbah\P\PRACTICAL 10.py'
Random Float number between 5 to 50: 19.340380948555122
PS D:\Misbah\P>
```