

# Python Revision Tour



# CHAPTER- 1

## PYTHON REVISION TOUR

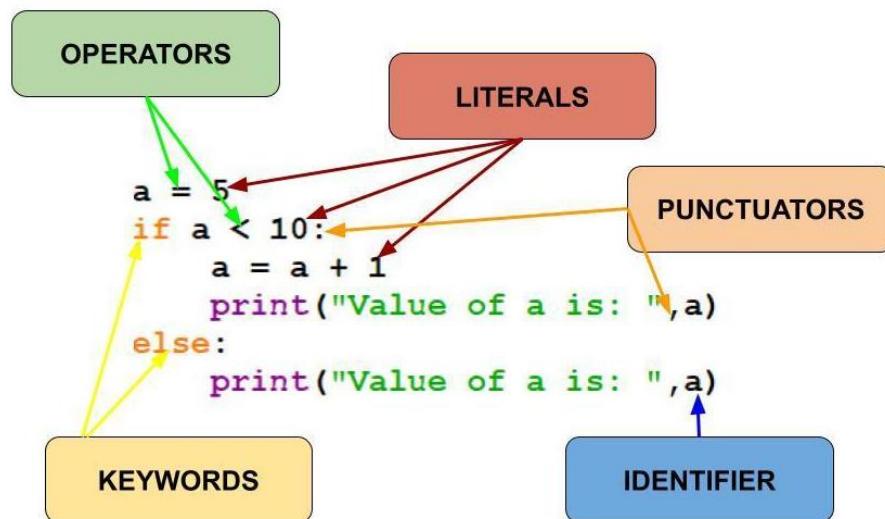
### ABOUT PYTHON:

1. Python is a high-level programming language developed by Guido Van Rossum. The language was released in February 1991 and got its name from a BBC comedy series “Monty Python’s Flying Circus”.
2. It is an interpreted and platform independent language i.e. the same code can run on any operating system.
3. It can be used to follow both procedural and object-oriented approaches to programming.
4. It is free to use and based on two programming languages: ABC language and Modula-3.



### BASIC TERMS USED IN PYTHON:

1. **Token / Lexical Unit:** The smallest individual unit in a python program is known as Token or Lexical Unit. A token has a specific meaning for a python interpreter. Examples of tokens are: Keywords, Identifiers, Literals, Operators and Punctuators.



2. **Keywords:** Keywords are the reserved words and have special meaning for Python Interpreters. Each keyword can be used only for that purpose which it has been assigned. Examples: and, while, del, with, True, None, False, return, try etc.
3. **Identifiers:** These are the names given to variables, objects, classes or functions etc. there are some predefined rules for forming identifiers which should be followed else the program will raise Syntax Error.

4. **Literals:** The data items that have a fixed value are called Literals. If a data item holds numeric values it will be known as Numeric Literal, if it contains String values it will be known as String Literal and so on. It means the type of value stored by the data item will decide the type of Literal. Python has one special literal which is None to indicate absence of value.
5. **Operators:** These are the symbols that perform specific operations on some variables. Operators operate on operands. Some operators require two operands and some require only one operand to operate. The operator precedence in Python is as follows:

Operator	Description	
( )	Parentheses (grouping)	Highest
**	Exponentiation	
~x	Bitwise nor	
+x,-x	Positive , negative (unary +,-)	
*,/,,%,//	Multiplication , division , floor division , remainder	
+-	Addition , subtraction	
&	Bitwise and	
^	Bitwise XOR	
	Bitwise OR	
<,<=,>,>=,<>,!=,==, is, isnot	Comparisons (Relational operators),identity operators	
not x	Boolean NOT	
And	Boolean AND	
or	Boolean OR	Lowest

**NOTE:** When we compare two variables pointing to same value, then both Equality (==) and identity (is) will return True. But when same value is assigned to different objects, then == operator will return True and is operator will return False.

6. **Punctuators:** These are the symbols that are used to organize sentence structure in programming languages. Common punctuators are: ‘ ’ # \$ @ [] {} = : ; () , .
7. **Variables:** In Python, variables are not storage containers like other programming languages. These are the temporary memory locations used to store values which will be used in the program further. Each time we assign a new value to a variable it will point to a new memory location where the assigned value is stored. In Python we do not specify the size and type of variable, besides these are decided as per the value we assign to that variable.
8. **Data Type:** It specifies the type of data we will store in the variable according to which memory will be allocated to that variable and it will also specify the type of operations that can be performed on that variable. Examples: integer, string, float, list etc.

**9. Dynamic Typing:** It means that it will be decided at the run time that which type of value the variable will store. It is also called implicit conversion. For example,

```
a = 'hello'    #a is of string type
a = 12         #a is of integer type
a = 12.56      # a is of float type
b = 10
c = a + b
a
12.56
b
10
c
22.560000000000002
```

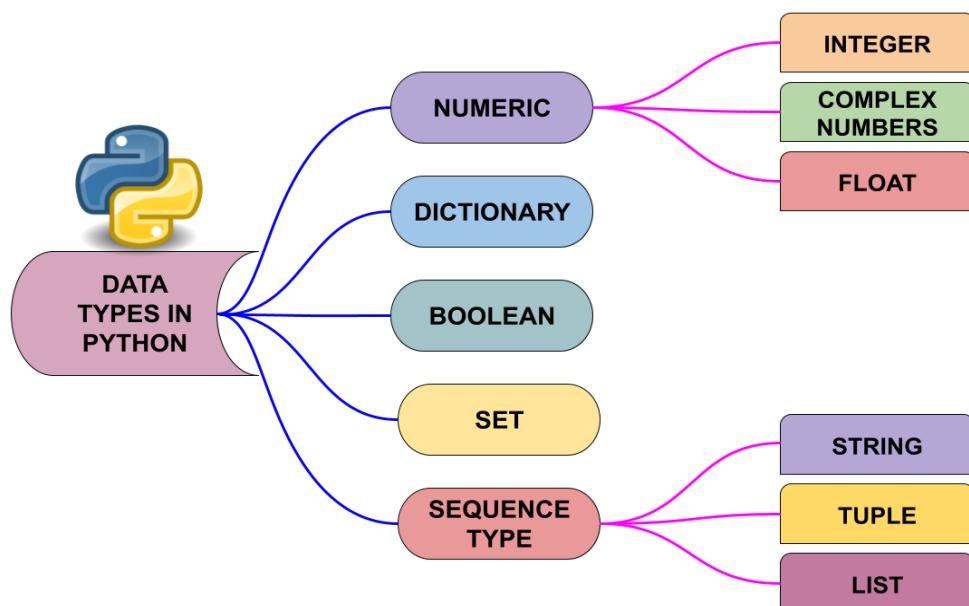
Here, we need not to specify the type of value a will store besides we can assign any type of value to a directly. Similarly, the data type of c will be decided at run time on the basis of value of a and b.

**10. Type Casting:** In Type casting, the data type conversion of a variable is done explicitly by using some built-in functions. Here, we can say that we force the variable by applying a built-in function to change the data type and it is not done at run time. Some common type casting functions are int(), float(), str(), list(), tuple(), dict() etc.

```
a = 12      #a is of integer type
b = float(a)  #changing a from integer to float
b          #now b will contain float value of a
12.0

t = (1,2,3,4)
s = list(t)
s
[1, 2, 3, 4]
```

## DATA TYPES IN PYTHON:

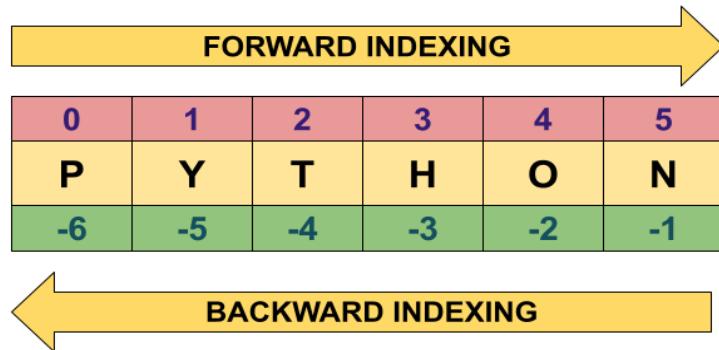


Above data types are classified in two basic categories: Mutable data types and Immutable data types. **Mutable data types** are those data types whose value can be changed without creating a new object. It means mutable data types hold a specific memory location and changes are made directly to that memory location. **Immutable data types** are those data types whose value cannot be changed after they are created. It means that if we make any change in the immutable data object then it will be assigned a new memory location.

1. In **Numeric data types**, Integers allow storing whole numbers only which can be positive or negative. Floating point numbers are used for storing numbers having fractional parts like temperature, area etc. In Python, Floating point numbers represent double precision i.e. 15-digit precision. Complex numbers are stored in python in the form of  $A + Bj$  where A is the real part and B is the imaginary part of complex numbers.
2. **Dictionary** is an unordered set of comma separated values where each value is a **key:value** pair. We represent the dictionary using curly brackets {}. Keys in the dictionary should be unique and they cannot be changed while values of the keys can be changed.
3. **Boolean** allows to store only two values True and False where True means 1 and False means 0 internally.
4. **Sequence data types** store a collection or set of values in an ordered manner. We can traverse the values/elements using indexing. The sequence data types are:

#### **A. STRING:**

- In Python, string is a sequence having Unicode characters. Any value written/assigned in “ ” or ‘ ’ is considered as a string in python.
- Each character is at a particular place called Index having starting value 0 if traversing forward. Indexing can also be done in backward direction starting from the last element/character of string where starting value will be -1 in backward direction.
- Strings are immutable.
- Joining operation in a string can be done between two strings only. We cannot join a number and a string using ‘+’.
- Concatenation operation in a string can be done between a string and a number only using ‘\*’.
- Slicing is defined as extracting a part of string from the main string using unique index positions starting from 0. In slicing we can specify start, stop and step values to extract substring from the given string.



## B. LIST:

- Lists can hold multiple elements of the same or different data types.
- Lists are mutable in nature which means the values can be updated without assigning a new memory location. It is denoted by square brackets [].
- ‘for’ loop can be used to traverse a list.
- We can add (join) a list with another list only and not with int, float or string type. Joining of 2 or more can be done using ‘+’.
- We can concatenate a list with an integer only. Concatenation (Replication) can be done in a list using ‘\*’.
- Slicing means extracting a part of list. Slicing in a list is done in same way as in String.

## C. TUPLE:

- Tuples can also hold multiple elements of the same or different data types like lists but tuples are immutable in nature. These are denoted by round brackets () .
- If multiple values are assigned to a single variable then by default the type of variable will be tuple.
- Tuples can be traversed in same way as Strings using ‘for’ loop.
- Slicing, Concatenation (Replication) and Joining operations can also be performed on Tuples in same way as of Strings.

## **WIDELY USED BUILT-IN FUNCTIONS IN PYTHON:**

### **STRING FUNCTIONS:**

<code>len (string)</code>	It returns the number of characters in any string including spaces.
<code>capitalize()</code>	It is used to convert the first letter of sentences in capital letter.
<code>title()</code>	It is used to convert first letter of every word in string in capital letters.
<code>upper()</code>	It is used to convert the entire string in capital case letters.
<code>lower()</code>	It is used to convert entire string in small case letters.
<code>count(substring, [start], [end])</code>	It is used to find the number of occurrences of substring in a string. We can also specify starting and ending index to specify a range for searching substring.
<code>find(substring, [start],[end])</code>	This function returns the starting index position of substring in the given string. Like <code>count()</code> , we can specify the range for searching using starting and ending index. <b>It returns -1 if substring not found.</b>
<code>index(substring)</code>	It returns the starting index position of substring. If substring not found then it will return an error " <b>Substring not found</b> ".
<code>isalnum()</code>	It is used to check if all the elements in the string are alphanumeric or not. It returns either True or False.
<code>islower()</code>	It returns True if all the elements in string are in lower case, otherwise returns False.
<code>isupper()</code>	It returns True if all the elements in string are in upper case, otherwise returns False.
<code>isspace()</code>	It returns True if all the elements in string are spaces, otherwise returns False.
<code>isalpha()</code>	It returns True if all the elements in string are alphabets, otherwise returns False.
<code>isdigit()</code>	It returns True if all the elements in string are digits, otherwise returns False.
<code>split([sep])</code>	This function is used to split the string based on delimiter/separator value which is space by default. It returns a list of n elements where the value of n is based on delimiter. The delimiter is not included in the output.
<code>partition(sep)</code>	It divides the string in three parts: head, separator and tail, based on the <b>sep</b> value which acts as a delimiter in this function. It will always return a tuple of 3 elements. The delimiter/separator will be included as 2 <sup>nd</sup> element of tuple in the output.
<code>replace(old, new)</code>	It is used to replace old substring inside the string with a new value.
<code>strip([chars])</code>	It returns a copy of string after removing leading and trailing white spaces by default. We can also provide <b>chars</b> value if we want to remove characters instead of spaces. If <b>chars</b> is given then all possible combination of given characters will be checked and removed.
<code>lstrip([chars])</code>	It returns a copy of string after removing leading white spaces. If <b>chars</b> value is given then characters will be removed.
<code>rstrip([chars])</code>	It returns a copy of string after removing trailing white spaces. If <b>chars</b> value is given then characters will be removed.

## **LIST FUNCTIONS:**

index()	Used to get the index of first matched item from the list. It returns index value of item to search. If item not found, it will return ValueError: n is not in the list.
append()	Used to add items to the end of the list. It will add the new item but not return any value
extend()	Used for adding multiple items. With extend we can add multiple elements but only in form of a list to any list. Even if we want to add single element it will be passed as an element of a list.
insert()	Used to add elements to list at position of our choice i.e. we can add new element anywhere in the list.
pop()	Used to remove item from list. It raises an exception if the list is already empty. By default, last item will be deleted from list. If index is provided then the given indexed value will be deleted.
remove()	Used to remove an element when index is not known and we want to delete by provided the element value itself. It will remove first occurrence of given item from list and return error if there is no such item in the list. It will not return any value.
clear()	Use to remove all the items of a list at once and list will become empty.
Del	del statement is used to delete the structure of existing list.
count()	Used to count the number of occurrences of the item we passed as argument. If item does not exist in list, it returns Zero.
reverse()	Used to reverse the items of a list. It made the changes in the original list and does not return anything.
sort()	Used to sort the items of a list. It made the changes in the original list and sort the items in increasing order by default. We can specify reverse argument as True to sort in decreasing order.
sorted()	Used to sort the items of a <b>sequence data type</b> and returns a list after sorting in increasing order by default. We can specify reverse argument as True to sort in decreasing order.

## **TUPLE FUNCTIONS:**

len()	Returns number of elements in a tuple
max()	Returns the element having maximum value in the tuple
min()	Returns the element having minimum value
index()	Returns index value of given element in the tuple. If item doesn't exist, it will raise ValueError exception.
count()	It returns the number of occurrences of the item passed as an argument. If not found, it returns Zero.

sorted()	Used to sort the items of a <b>sequence data type</b> and returns a list after sorting in increasing order by default. We can specify reverse argument as True to sort in decreasing order.
----------	---

### **Dictionary Functions:**

clear()	Used to remove all items from dictionary
get()	Used to access the value of given key, if key not found it raises an exception.
items()	Used to return all the items of a dictionary in form of tuples.
keys()	Used to return all the keys in the dictionary as a sequence of keys.
values()	Used to return all the values in the dictionary as a sequence of values.
update()	Merges the key:value pair from the new dictionary into original dictionary. The key:value pairs will be added to the original dictionary, if any key already exists, the new value will be updated for that key.
fromkeys()	Returns new dictionary with the given set of elements as the keys of the dictionary.
copy()	It will create a copy of dictionary.
popitem()	Used to remove the last added dictionary item (key:value pair)
max()	Used to return highest value in dictionary, this will work only if all the values in dictionary are of numeric type.
min()	Used to return lowest value in dictionary, this will work only if all the values in dictionary are of numeric type.
sorted()	Used to sort the key:value pair of dictionary in either ascending or descending order based on the keys.

### **Statements in Python:**

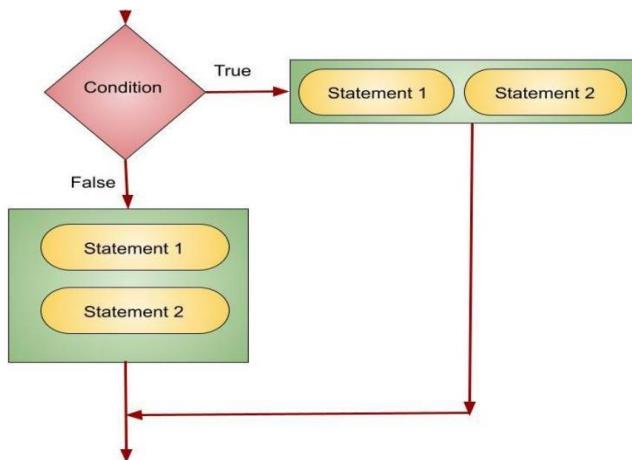
Instructions given to computer to perform any task are called Statements. In Python, we have 3 type of statements:

- **EMPTY STATEMENTS:** When a statement is required as per syntax but we don't want to execute anything or do not want to take any action we use *pass* keyword. Whenever *pass* is encountered, python interpreter will do nothing and control will move to the next statement in flow of control.
- **SIMPLE STATEMENT:** All the single executable statements in Python are Simple Statements.
- **COMPOUND STATEMENTS:** A group of statements executed as a unit are called compound statements. Compound statements has a Header which begins with a

keyword and ends with colon (:). There can be at least one or more statements in the body of the compound statement, all indented at same level.

## Conditional statements in Python:

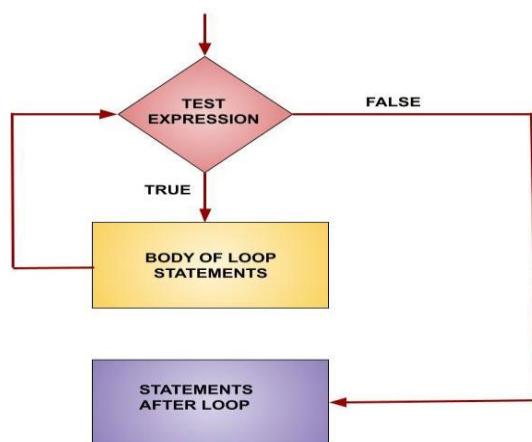
When the execution of any statement depends on some condition then such statements are considered as Conditional Statements. In Python, we use *if* keyword for Conditional statements.



- ✚ It must contain valid condition which evaluates to either True or False.
- ✚ Condition must follow by Colon (:), it is mandatory.
- ✚ Statement inside if must be at same indentation level.
- ✚ if statement can be of many forms:
  - if without false statement
  - if with else
  - if with elif
  - Nested if

## Iterative statements in Python:

In Python, to perform repetition we have two keywords, *for* (Counting Loop) and *while* (Conditional Loop)



- *for* loop in python is used when we know the number of iterations.
- *for* loop is used to create loop where we are working on sequence data types.
- To repeat the loop **n number of times** in *for* loop we use range function in which we can specify lower limit and upper limit.
- **range function** will generate set of values from lower limit to upper limit. We can also specify the step value which is +1 by default.
- Step value can be in -ve also to generate set of numbers in reverse orders.
- *while* loop in python is also called as entry-controlled loop in which entry is loop is allowed only if the condition is true.
- There are 4 main elements of *while* loop:
  - **Initialization:** In while loop, we cannot use a variable in test condition without initializing it with a starting value. So, we will specify the starting value to the variable to be used in loop.
  - **Test Condition:** We will specify a test condition in the header of the loop. If this condition will be True then only the body of the loop will be executed else the loop will not get executed at all.
  - **Body of loop:** We will write the statements to be executed if the test condition will be True.
  - **Update Statement:** In while loop, we have to increase and decrease the value of variable used in test condition else the loop will result in infinite loop.
- **break** keyword is used to take control out of the loop for any given condition. When *break* is encountered in a loop the flow of control jumps to the very next statement after the loop.
- **continue** keyword is used to skip the execution of remaining statements inside the loop and takes control to next iteration..

## **MULTIPLE CHOICE QUESTION**

- 1 Find the invalid identifier from the following
  - a) None
  - b) address
  - c) Name
  - d) pass
- 2 Write the type of tokens from the following:
  - a) If
  - b) roll\_no
- 3 Consider a declaration L = (1, 'Python', '3.14').  
Which of the following represents the data type of L?
  - a) List
  - b) Tuple
  - c) Dictionary
  - d) String
- 4 Identify the valid arithmetic operator in Python from the following.
  - a) ?
  - b) <
  - c) \*\*
  - d) And
- 5 Which of the following statements is/are not python keywords?
  - a) False
  - b) Math
  - c) WHILE
  - d) Break
- 6 Which of the following is a valid keyword in Python?
  - a) False
  - b) return
  - c) non\_local
  - d) none
- 7 State True or False.  
"Identifiers are names used to identify a variable, function in a program".
- 8 Identify the invalid identifier out of the options given below.
  - a) Qwer\_12
  - b) IF
  - c) Play123
  - d) Turn.over
- 9 One of the following statements will raise error. Identify the statement that will raise the error.

- a) `>>>| x,y=20`  
b) `|>>>| a,b=6, 7*9`  
c) `>>>| a=b=c=35`  
d) None of above will raise error
- 10 Given the following Tuple  
`Tup (10, 20, 30, 50)`  
Which of the following statements will result in an error?  
a) `print (Tup [0])`  
b) `print (Tup [1:2])`  
c) `Tup.insert (2,3)`  
d) `print(len (Tup))`
- 11 Consider the given expression : 5<10 and 12>7 or not 7>4  
Which of the following will be the correct output, if the given expression is evaluated?  
a) True  
b) False  
c) NULL  
d) NONE
- 12 Which of the following will give output as [5,14,6] if `lst=[1,5,9,14,2,6]`?  
a) `print(lst[0::2])`  
b) `print(lst[1::2])`  
c) `print(lst[1:5:2])`  
d) `print(lst[0:6:2])`
- 13 The return type of the `input()` function is  
a) string  
b) Integer  
c) list  
d) tuple
- 14 Which of the following operator cannot be used with string data type?  
a) +  
b) In  
c) \*  
d) /
- 15 Consider a tuple `tup1 = (10, 15, 25, and 30)`. Identify the statement that will result in an error.  
a) `print(tup1[2])`  
b) `tup1[2] = 20`  
c) `print(min(tup1))`  
d) `print(len(tup1))`
- 16 Which one of the following is the default extension of a Python file?  
a) .exe  
b) .p++

- c) .py  
d) .p
- 17 Which of the following symbol is used in Python for single line comment?  
a) /  
b) /\*  
c) //  
d) #
- 18 Which of these about a dictionary is false?  
a) The values of a dictionary can be accessed using keys  
b) The keys of a dictionary can be accessed using values  
c) Dictionaries aren't ordered  
d) Dictionaries are mutable
- 19 Which is the correct form of declaration of dictionary?  
a) Day={1:'monday',2:'tuesday',3:'wednesday'}  
b) Day=(1;'monday',2;'tuesday',3;'wednesday')  
c) Day=[1:'monday',2:'tuesday',3:'wednesday']  
d) Day={1'monday',2'tuesday',3'wednesday'}
- 20 What will be the output of the following statement:  
`print(3-2**2**3+99/11)`  
a) 244  
b) 244.0  
c) -244.0  
d) Error
- 21 What is the output of following code:  
`T=(100) print(T*2)`  
a) Syntax error  
b) (200,)  
c) 200  
d) (100,100)
- 22 Identify the output of the following Python statements:  
`x = [[10.0, 11.0, 12.0],[13.0, 14.0, 15.0]] y = x[1][2] print(y)`  
a) 12.0  
b) 13.0  
c) 14.0  
d) 15.0
- 23 Select the correct output of the code :  
`S= "Amrit Mahotsav @ 75" A=S.partition (" ") print (a)`  
a) ('Amrit Mahotsav', '@', '75')  
b) ['Amrit', 'Mahotsav', '@', '75']  
c) ('Amrit', 'Mahotsav @ 75')  
d) ('Amrit', "", 'Mahotsav @ 75')

24 Identify the output of the following Python statements.

```
x = 2
while x < 9:
    print(x, end="")
    x = x + 1
a) 12345678
b) 123456789
c) 2345678
d) 23456789
```

25 Identify the output of the following Python statements.

```
b = 1
for a in range(1, 10, 2):
    b += a + 2
    print(b)
a) 31
b) 33
c) 36
d) 39
```

26 A tuple is declared as T = (2,5,6,9,8). What will be the value of sum(T)?

27 Identify the output of the following Python statements.

```
lst1 = [10, 15, 20, 25, 30]
lst1.insert( 3, 4)
lst1.insert( 2, 3)
print (lst1[-5])
a) 2
b) 3
c) 4
d) 20
```

28 Evaluate the following expression and identify the correct answer.

$$16 - (4 + 2) * 5 + 2**3 * 4$$

a) 54  
b) 46  
c) 18  
d) 32

29 Fill in the blank.

\_\_\_\_\_ function is used to arrange the elements of a list in ascending order.

- a) sort()
- b) ascending()
- c) arrange()
- d) asort()

- 30 Which of the following will delete key-value pair for key = "Red" from a dictionary D1?
- a) delete D1("Red")
  - b) del D1["Red"]
  - c) del.D1["Red"]
  - d) D1.del["Red"]
- 31 Identify the valid declaration of L: L = ['Mon', '23', 'hello', '60.5']
- a) dictionary
  - b) string
  - c) tuple
  - d) list
- 32 Given a Tuple tup1= (10, 20, 30, 40, 50, 60, 70, 80, 90). What will be the output of print (tup1 [3:7:2])?
- a) (40,50,60,70,80)
  - b) (40,50,60,70)
  - c) [40,60]
  - d) (40,60)
- 33 If the following code is executed, what will be the output of the following code?  
name="ComputerSciencewithPython"  
print(name[3:10])
- 34 Which of the following statement(s) would give an error during execution of the following code?
- ```
tup = (20, 30, 40, 50, 80, 79)

print(tup)           #Statement 1
print(tup[3]+50)    #Statement 2
print(max(tup))     #Statement 3
tup[4]=80          #Statement 4
```
- a) Statement 1
  - b) Statement 2
  - c) Statement 3
  - d) Statement 4
- 35 Consider the statements given below and then choose the correct output from the given options:  
pride="#G20 Presidency"  
print(pride[-2:2:-2])
- a) ndsr
  - b) ceieP0
  - c) ceieP
  - d) yndsr
- 36 Given is a Python list declaration :  
Listofnames=["Aman", "Ankit", "Ashish", "Rajan", "Rajat"]

- Write the output of:  
`print (Listofnames [-1:-4:-1])`
- 37 What will be the output of the following code?  
`Cities=['Delhi','Mumbai']  
Cities[0],Cities[1]=Cities[1],Cities[0]  
print(Cities)`
- 38 What will be the output of the following code?  
`tup1 = (1,2,[1,2],3)  
tup1[2][1]=3.14  
print(tup1)`  
a) (1,2,[3.14,2],3)  
b) (1,2,[1,3.14],3)  
c) (1,2,[1,2],3.14)  
d) Error Message
- 39 Select the correct output of the code:  
`s = "Python is fun"  
l = s.split()  
s_new = "-".join([l[0].upper(), l[1], l[2].capitalize()])  
print(s_new)`  
a) PYTHON-IS-Fun  
b) PYTHON-is-Fun  
c) Python-is-fun  
d) PYTHON-Is –Fun
- 40 What will be the result of following python code?  
`a,b=5,10  
a,b=b+2,a-1  
print("a=",a,"b=",b)  
a+=2  
b=b/2  
print("a=",a,"b=",b)`
- 41 For a string S declared as S="PYTHON", Which of the following is incorrect:  
a) N=len(s)  
b) T=S  
c) "T" in S  
d) S[0]="M"
- 42 Which of the following statement(s) would give an error after executing the following code?  
`Stud={"Murugan":100,"Mithu":95} # Statement 1  
print (Stud [95]) # Statement 2  
Stud ["Murugan"]=99 # Statement 3  
print (Stud.pop()) # Statement 4`

- ```
print (Stud)                      # Statement 5
    a) Statement 2
    b) Statement 4
    c) Statement 3
    d) Statements 2 and 4
```
- 43 What will be the output of following Python code?
- ```
>>> d={1:"Ajay",2:"Neeta",3:"Saira"}
...
>>> print("Neeta" in d)
```
- a) False  
b) True  
c) Error  
d) None
- 44 Write a statement in Python to declare a dictionary whose keys are 1, 2, 3 and values are Monday, Tuesday and Wednesday respectively.
- 45 Assertion(A): List is an immutable data type  
Reasoning(R): When an attempt is made to update the value of an immutable variable, the old variable is destroyed and a new variable is created by the same name in memory
- a) Both A and R are true and R is the correct explanation for A  
b) Both A and R are true and R is not the correct explanation for A  
c) A is True but R is False  
d) A is false but R is True
- 46 Assertion (A): Python Standard Library consists of various modules.  
Reasoning(R): A function in a module is used to simplify the code and avoids repetition.
- a) Both A and R are true and R is the correct explanation for A  
b) Both A and R are true and R is not the correct explanation for A  
c) A is True but R is False  
d) A is false but R is True
- 47 Assertion(A): List is an immutable data type  
Reasoning(R): When an attempt is made to update the value of an immutable variable, the old variable is destroyed and a new variable is created by the same name in memory.
- a) Both A and R are true and R is the correct explanation for A  
b) Both A and R are true and R is not the correct explanation for A  
c) A is True but R is False  
d) A is false but R is True
- 48 Assertion (A) : List can not become key in a dictionary.  
Reasoning(R) : Only integers can be keys in a dictionary.

- a) Both A and R are true and R is correct explanation of A
- b) Both A and R are true but R is not correct explanation of A
- c) A is True but R is false
- d) R is true but A is false

### **Weightage : 2 marks**

- 1 Rewrite the following code in python after removing all syntax error(s). Underline each correction done in the code.

```
30=To
for K in range(0,To)
    IF k%4==0:
        print (K*4)
    Else:
        print (K+3)
```

- 2 Rewrite the following code after removing all syntax error(s) and underline each correction done:

```
Runs=(10,5,0,2,4,3)
for I in Runs:
    if I=0:
        print(Maiden Over)
    else:
        print(Not Maiden)
```

- 3 Evaluate the following Python expression:

- a)  $2*3+4^{**}2-5//2$
- b)  $6<12$  and not  $(20>15)$  or  $(10>5)$

- 4 Predict the output of the following code:

```
S="LOST"
L=[10,21,33,4]
D={}
for I in range(len(S)):
    if I%2==0:
        D[L.pop()]=S[I]
    else:
        D[L.pop()]=I+3
for K,V in D.items():
    print(K,V,sep="*")
```

- 5 Write the Python statement for each of the following tasks using BUILT-IN functions/methods only:

- (i) To insert an element 200 at the third position, in the list L1.
- (ii) To check whether a string named, message ends with a full stop / period or not.

- 6 A list named studentAge stores age of students of a class. Write the Python command to import the required module and (using built-in function) to display the most common age value from the given list.

## Weightage : 3 marks

- 1 Find the output of the following code:

```
Name="PythoN3.1"
R=""
for x in range(len(Name)):
    if Name[x].isupper():
        R=R+Name[x].lower()
    elif Name[x].islower():
        R=R+Name[x].upper()
    elif Name[x].isdigit():
        R=R+Name[x-1]
    else:
        R=R+"#"
print(R)
```

- 2 Predict the output of the Python code given below:

```
Text1="IND-23"
Text2=""
I=0
while I<len(Text1):
    if Text1[I]>="0" and Text1[I]<="9":
        Val = int(Text1[I])
        Val = Val + 1
        Text2=Text2 + str(Val)
    elif Text1[I]>="A" and Text1[I]<="Z":
        Text2=Text2 + (Text1[I+1])
    else:
        Text2=Text2 + "*"
    I+=1
print(Text2)
```

## Programming Practice

- 1 Write a function, lenWords(STRING), that takes a string as an argument and returns a tuple containing length of each word of a string. For example, if the string is "Come let us have some fun", the tuple will have (4, 3, 2, 4, 4, 3)
- 2 Write a function countNow(PLACES) in Python, that takes the dictionary, PLACES as an argument and displays the names (in uppercase)of the places whose names are longer than 5 characters. For example, Consider the following dictionary

```
PLACES={1:"Delhi",2:"London",3:"Paris",4:"New York",5:"Doha"}
```

The output should be:

LONDON NEW YORK

- 3 Write a function LShift(Arr,n) in Python, which accepts a list Arr of numbers and n is a numeric value by which all elements of the list are shifted to left. Sample Input

**Data of the list**

Arr= [ 10,20,30,40,12,11],

n=2

**Output:**

Arr = [30,40,12,11,10,20]