STAR SHINE

Date / /

Page

Python

Inventor of Python = Guido Van Rossum in Feb 1991.

The name of Python Programming language comes from an old BBC television comedy sketch series called Monty Python's Flying Circus.

Python Arithematic Operators:

These are the operators that are used to perform mathematical operations on numerical values.

There are 7 types of operators:

1	+	Addition
2	−	Subtraction
3	*	Multiplication
4	/	Division
5	%	Modulus
6	//	Floor division
7	**	Exponent Power

Comments:

Comments in python are the lines in the code that are ignored by the interpreter during the execution of the program.

Teacher Signature .....

Parents Signature .....

Date / /
Page

→ they are used to explain code

Comments enhance the readability of the code and help the programmers to understand the code carefully.

Types of Comments:

1. Single line Comment
2. Multiline Comment
3. String literals Comment.

- Single line Comment: Python single line comment starts with hashtag (#) symbol with no white space.

- Multiline Comment: We can use multiple hashtags to write multiline comments in python.

eg: #Python Program
#Demonstrate Comment.

- String literals: Any string enclosed within double or single quotes is considered a string literal and is ignored by interpreter if it is not assigned to a variable

a. Single line: Here we use single quote.

eg: 'This is comment'

b. Multiline: Here we use triple double quote.

eg: """This is comment
Yes, this is"""

Teacher Signature .....

Parents Signature .....

STAR SHINE

Date / /

Page:

Variables in Python:

These are the containers that store values. In this we do not need to declare the variable or declare their type before using them. Python variable is a name given to a memory location.

It is a basic unit of storage in a program.

A value holding python variable is also known as "Identifier".

Rules:

1. A python variable name must start with a letter or a underscore character.
2. It cannot start with a number.
3. It can only contain alpha numeric characters and underscore [(A-Z)(0-9)(_)].
4. Variables in python are case sensitive.
5. The keyword in python cannot be used to name the variable.
6. We cannot use any special symbol and cannot put white space.

eg: Ram_94, B_h_g, B985, Z_91 etc.

Teacher Signature .....

Parents Signature .....

Date / /
Page

Rules for multiword variable.

1. Camel Case ⟶ nameOfStudent
2. Pascal Case ⟶ NameOfStudent
3. Snake Case ⟶ name_of_student

Python assign value to multiple variables:

```
a=b=c=10
print(a)
print(b)
print(c)
```

```
a,b,c = 10,15.5,"abc"
print(a)
print(b)
print(c)
```

Local Variables:

These are those which are initialized inside a function and belong only to that particular function. It cannot be accessed anywhere outside the function.

eg:
```
def f():
    s = 'abc'
    print(s)
f()
```

Global Variables:

These are those which are defined outside any function and which are accessible throughout

Teacher Signature ..... Parents Signature .....

the program, i.e, inside and outside of every function.

eg: def f():
    print(s)
    s = 'abc'
f()

Note: Deleting a variable in python del keyword is used.

eg: del Variable_name

Assignment Operators:

Python has assignment operator that helps to assign values or expressions to the left hand side variable. It is represented by (=) equals to symbol.

Types of assignment operators:

1. += Addition assignment
2. -= Subtraction assignment
3. /= Division assignment
4. *= Multiplication assignment
5. %= Modulus assignment
6. //= Floor division assignment
7. **= Exponentiation assignment

8. &= Bitwise And assignment
9. ^= Bitwise XOR assignment
10. |= Bitwise OR assignment
11. >>= Bitwise Right Shift assignment
12. <<= Bitwise Left Shift assignment

Examples: a = 6
b = 13
a &= b ⟶ a = a & b
print(a)
Output: 4

a	b	R	a & b
0	0	0	↓
0	1	0	0 1 1 0
1	0	0	1 1 0 1
1	1	1	0 1 0 0 ⇒ 4

Example: a = 6
b = 13
a |= b ⟶ a = a | b
print(a)
Output: 15

a	b	R	a | b
0	0	0	↓
0	1	1	0 1 1 0
1	0	1	1 1 0 1
1	1	1	1 1 1 1 ⇒ 15

Example: a = 6
b = 13
a ^= b ⟶ a = a ^ b
print(a)
Output: 11

a	b	R	a ^ b
0	0	0	↓
0	1	1	0 1 1 0
1	0	1	1 1 0 1
1	1	0	1 0 1 1 ⇒ 11

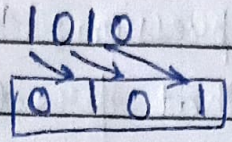
Teacher Signature .....

Parents Signature .....

Date / /
Page

Example a = 10

a >> 1 → a = a >> 1



print (a)

Output: 5

Example a = 10

a << 1 → a = a << 1

print (a)

Output: 20

Expressions:

Expressions is a combination of operators & operands that is interpreted to produce some other values. An expression is evaluated as per the precedence of its operators.

Types of Expressions:

1. Constant Expression: These are the expressions that have constant value only. eg: x = 15 + 2 * 3.

2. Arithematic Expression: It is the combination of numeric value and some times parenthesis (brackets). The result of this expression is also a numeric value. eg: +, -, *, / etc.

Teacher Signature .....

Parents Signature .....

Date / /
Page

3 Integral Expression: These are the kind of expression that produce only integer results after all compititions and type conversion.
eg: a=10 b=12.5 print (int(a+b)) output = 22

4. Floating Expression: These kind of expressions always produce floating point number as result.

5. Relational Expression: In these types of expression arithematic operations are written on both sides of relational operators.
eg: print ((a+b)>=(c-d))
Those arithematic expressions are evaluated first and then compared as per relational operator and produce a boolean output in the end.

6. Logical Expression: It produce boolean output in the end. Here we use logical operator which are AND, OR and NOT.

7. Bitwise Expression: These are the kind of expression in which computations are performed at bit level. These are expression in which different bitwise are used.

Teacher Signature .....

Parents Signature .....

Here we use diff. type of expressions in a single expression.

8. Combinational Expression: Print (a+(b>>1))

9. Multi-Operator expression: → There are more than 1 operator.

a = 10+(4*3)	Output
print (a)	22
a = (10+4)*3	
print (a)	42
a = (10+4*3)	
print (a)	22

⇨ Precedence chart:

1 Paranthesis [( ), { }, [ ]]
2 **
3 +a, -a, ~a → complement.
4 /, *, //, %
5 +, -
6 >>, <<, [shift operation]
7 Bituise And
8 ^
9 |
10 >, <, >=, <=
11 ==, !=
12 Assignment operation [+=, -=, *=]
13 is, is not
14 And, Or, Not

Teacher Signature .....

Parents Signature .....

Date / /
Page

⇨ Datatype:

It identify the type of data that a variable can store. Variable stores different type of values hence the data type decides the type of value. Every value in python belongs to different data type. It represents the kind of value that tell what operations can be performed of particular data.

Built in data types in python:

Data Type	'Str'
Numeric Type	int, float, complex
Sequence Type	List, Tuple, range
Maping Type	Dict → Dictionary
Set Type	Set, frozen set
Boolean Type	bool
None Type	None

Note: To define the values of various data types and to check their data type we use 'type' function.

* Strings in python: String is the collection of one or more character put in a single quote, double quote.

Teacher Signature .....

Parents Signature .....

Date / /
Page

* In python their is no character data type. A character is a string of length 1 (one).

In python programming individual characters of a string can be accessed by using the method of indexing.

String Slicing:
eg: a='accessing'
0 1 2 3 4 5 6 7 8 9
We have to print sing from it.
∴ print [initial index : Final index+1]
∴ print(a[5:9])

- Numeric data type: It represent the data that has numeric value. It can be integer no, floating no., complex no.

→ Integer: It contain +ve or -ve no. There is no fraction or decimal no.
In python 3.x there is no limit to how long an integer value can be.

→ Float: It contain decimal numbers.

→ Complex: It contain real and imaginary part. eg: 3+4j

Teacher Signature .....

Parents Signature .....

Syntax of range- range(start, end, step)

- Sequence type :

→ Range :

eg: x = range(6)
print (x) Output : ~~range(0,6)~~ range(0,6)

eg: Print the counting from 1 to 10 using for.
for in in range (1,11):
↔ print (i) Output: 1
2
3
4
5
6
7
8
9
10

→ List : It is just like array. It is an order to collect data. It is very flexible as the items in the list do not need to be same type. It is list by just placing the sequences inside the square brackets.

eg: l1 = [ ] Outputs'
print (l1) [ ]
l2 = ['Muskan']
print (l2) ['Muskan']
l3 = ['Muskan', 'xyz']
print (l3) ['Muskan', 'xyz']

Teacher Signature .....

Parents Signature .....

Date / /
Page:

l4 = ['xyz', 1, 2.0]
print (l4) ['xyz', 1, 2.0]

Multi dimensional list (list of list)

eg l5 [['xyz', 1, 2.0], ['abc']]
print (l5 [1][0]) → abc

Access the element from the list.

l4 = ['xyz', 1, 2.0]
0 1 2
-3 -2 -1

print (l4 [0]) → xyz

- **Boolean type:** It is the datatype with one of the two built in values True or False. It is denoted by class bool

Ques Difference between List and Tuple.

Ans

List	Tuple
1. In list we use square brackets [ ].	1. In tuple we use normal (round) brackets ( ).
2. List is mutable that mean it can change.	2. Tuple is immutable that mean it cannot change.
3. Consumes more memory	3. Consume less memory.
4. They are more error prone.	4. Tuple operations are safe
5. In dic. we cannot create keys using list.	5. Create keys using tuples

Teacher Signature .....

Parents Signature .....

Date / /
Page

- **Tuple:** Tuples are just like list which is the ordered collection of elements. Tuples are created by placing the sequence of values separated by comma with the use of round brackets. It can contain the mixed datatype elements

Maping Type:

Dictionary: Dictionary in python is an unorganized collection of data value.
other data types hold only single value as element but dictionary hold a key value pair. It is mutable.
Key value pair in dictionary is separated by (:) colon, where as each key is separated by (,) comma.
We create the dictionary a (& 3) curly bracket.
Dictionary key are case sensitive, the same name but different cases of key will be treated differently.
Values in dictionary can be of any data type or can be duplicated.

Teacher Signature .....

Parents Signature .....

Date / /
Page:

Ques Ans Difference between mutable and Immutable

Mutable	Immutable.
• A mutable data type is those whose value can be changed.	An Immutable data type is one in which the values can't be changed or altered.
• Generally provide a method to add or remove element.	Does not provide method to add, remove or change.
• Slower to access.	Quicker to access.
• eg - Set, List, dict.	eg - String, frozen set, tuples.

* **Set:** It is a collection of unordered elements - that is mutable. Set has no duplicate value. Set can be of any data type. Example → String, Boolean, tuple, Int.

```
S1 = {'abc', 'xyz'}
S2 = {1, 2, 3}
S3 = {1.0, 2.0}
S4 = {1.0, 'abc', 2}
S5 = {1, 2, 3, 4, 3, 2}
l1 = set([1, 2, 3])
print(len[S4]) → to print the length.
S2.discard('abc') → delete
```

To add new element
S2.add(4)

Teacher Signature .....

Parents Signature .....

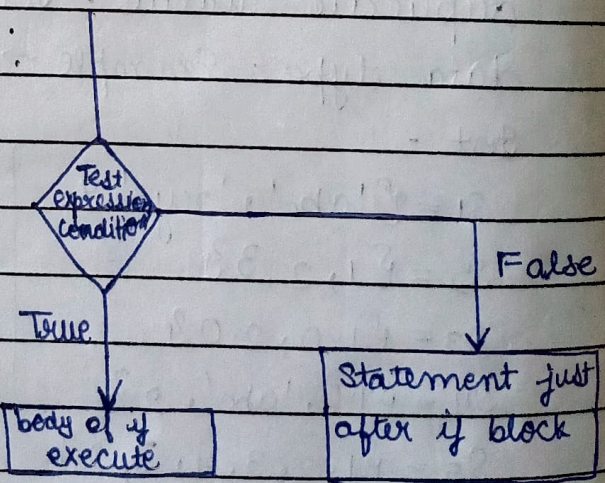
Date / /
Page

- Frozen Set → Immutable version of set
  Syntax: frozenset (iterable)
  ↳ string, list, Tuple, Dict, numbers.

⇒ conditions are used to control the flow of a program based on certain criteria. conditions are typically expressed through conditional statements which allow you to execute different block of a code depending on whether a condition is true or false.

Python 'if' statement
Syntax: if condition:
    # statement to execute
    # if the condition is true.

Flow Chart of 'if':



Teacher Signature ..... Parents Signature .....

Date / /
Page:

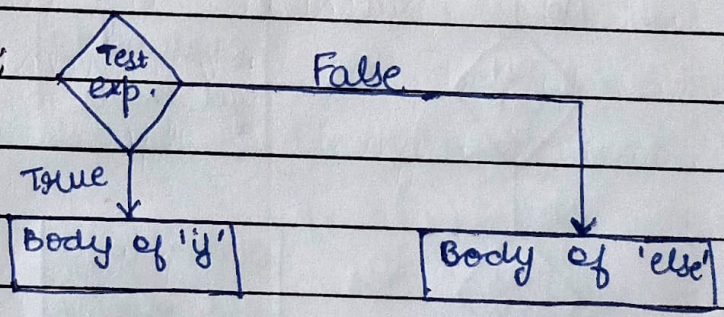
Python 'if else' Statement :

Syntax : if (condition):
    # Statement to execute
    # if condition is true.
else:
    # Statement to execute
    # if condition is false

- Program for if else statement:

```
num = int (input ("enter number"))
if (num > 0):
    print ("+ve no.")
else:
    print ("-ve no.")
```

Flow chart :



elif Statement: Syntax : if (condition):
    # Statement
elif (condition):
    # Statement
else:
    # Statement

Teacher Signature .....

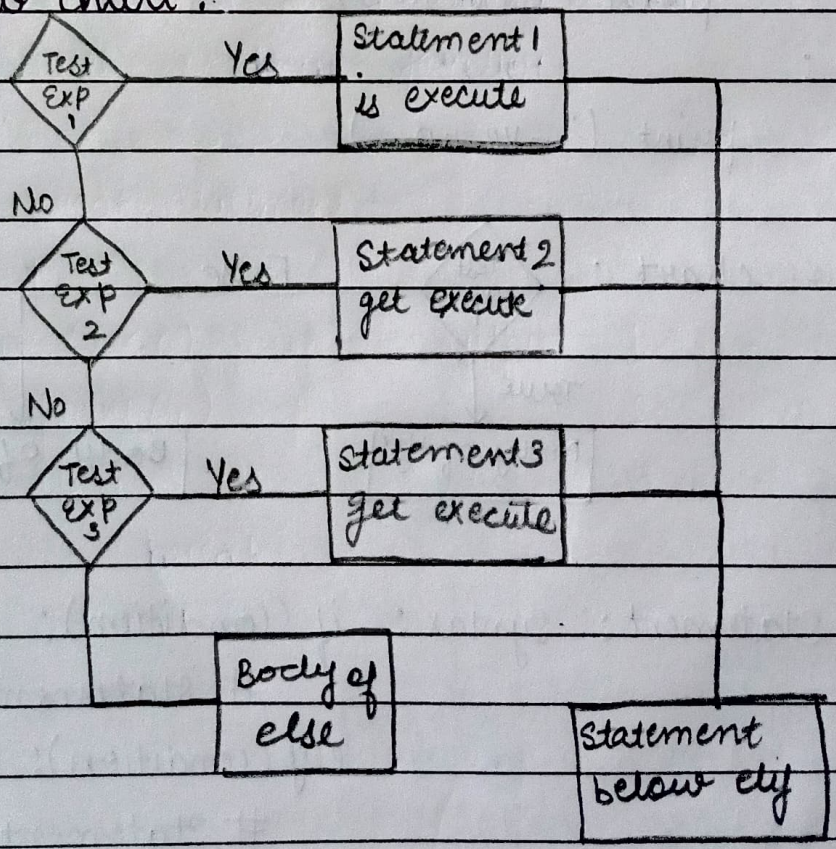
Parents Signature .....

Date / /
Page.

Program for elif Statement:

```
i = 20
if (i==0):
    print("i is 10")
elif (i==15):
    print("i is 15")
elif (i==20)
    print("i is 20")
else:
    print("i is not present")
```

Flow Chart:



Teacher Signature .....

Parents Signature .....

STAR SHINE

Date / /

Page:

- **Python loops:** Python has two primitive loop commands.

1. While loop
2. For loop

- **While loop:** In while loop, we can execute a set of statements as long as a condition is true.

Syntax: While (condition):

    # Statement

Example -
```
Count = 0
While (Count < 3):
    Count = Count + 1
    print ("Hello")
```
Output - Hello
Hello
Hello

- Else with while loop -

```
Count = 0
While (Count < 3):
    Count = Count + 1
    print ("Hello")
else:
    print ("Muskan")
```

Output - Hello
Hello
Hello
Muskan

Teacher Signature .....

Parents Signature .....

Date / /
Page

- Count = 8

```
while (count < 5):
        count = count + 1
        print ("Hello")
else:
        print ("Muskan")
```

Output - Muskan

- Count = 0

```
while (count == 0)
    print ("Hello")
```

Output - Hello
↓
infinite

Note: Remember to increment i, or else the loop will continue forever.

- The break statement -

```
i = 1
while (i < 6):
    print (i)
    if (i == 3):
        break
    i = i + 1
```

Output - 1
2
3

Teacher Signature .....

Parents Signature .....

Date / /
Page:

- The Continue statement

```
i=0
while (i<6):
    i=i+1
    if (i==3):
        continue
    print(i)
```

output - 1
2
4
5
6

- For loop: A for loop is used to iterating over a sequence (that is either a list, a tuple, a dict., a set or a string.)

Example - `l1 = ["abc", "xyz"]`

```
for i in l1:
    print(i)
```

#abc
#xyz

- `tuple1 = ("abc", "xyz")`

```
for i in tuple1:
    print(i)
```

abc
xyz

- `String1 = 'abc'`

```
for i in String1:
    print(i)
```

#a
#b
#c

Teacher Signature .....

Parents Signature .....

Date / /
Page

- Set1 = {1, 2, 3}
  for i in Set1 :
      print (i)

  #1
  #2
  #3

- list1 = ['abc', 'xyz']
  for i in range (len(list1)):
      print (list[i])

  # abc
  # xyz

- for i in range (3):
      print ("Hello")

  # Hello
  # Hello
  # Hello

- for i in range (3):
      print (i)
  else:
      print ("Muskan")

  0
  1
  2
  Muskan

Ques Difference between for loop and while loop.

Ans

For loop	While loop
• It is used when the no. of iterations is known.	It is used when the no. of iterations is not known.
• In case of no condition, the loop is repeated infinite times.	In case of no condition an error will be shown.

Teacher Signature .....

Parents Signature .....

Date / /
Page

For loop	While loop
• Initialization is not repeated.	Initialization is repeated if carried out during the stage of checking.
• Statement of iteration is written after running.	It can be written at any place.
• Initialization can be in or out of the loop	Initialization is always out of the loop.
• The nature of the increment is simple.	The nature of the increment is complex.
• Used when initialization is simple.	Used when initialization is complex.

⇨ Understanding error message in python:

Errors are the problems in a program due to which the program will stop the execution.

Three main types of error:

1. Syntax error
2. Logical error
3. Runtime error

Teacher Signature .....

Parents Signature .....

Date / /
Page

Syntax error: These occur when the code violates the rules of python language syntax. These errors are detected by the python interpreter when it is executing the code.

Runtime error: These occur when the code is syntactically correct but causes an error when it is executed. These errors can be caused by a variety of reasons such as invalid input data, division by zero, or excessing a undefined variable

Logical error: These occur when the code is syntatically correct and runs without causing any error but the output is not what you expected. These errors can be caused by incorrect logic of program or due to mathematical error.

Types of run time error:

1. Index error: eg :⇒ `l1 = [1, 2, 3]`
   `print(l1[3])`
2. Name error: eg → `a = 3`
   `print(b)`

Teacher Signature .....

Parents Signature .....

Date / /
Page:

3 Type error: eg → x = "20" → String
y = 10 → Int
print (x+y)

4. Attribute error: Jo cheez python ch exist ne krdi
eg → str1 = 'Hello'
print (str1.reverse())

5. Key error: eg → D1 = {1: 'ab', 2: 'xy'}
print (D1[3])

6. Stop iteration: eg → it = iter ([1,2,3])
print (next(it))
print (next(it))
print (next(it))
print (next(it))

Syntax / Compile time error:
Indentation error: eg → for i in range(5):
print (i)

Run time error
Divide by zero: eg → x = 5
print (x/0)

Teacher Signature .....

Parents Signature .....