

Python Keywords in Hindi - Python in Hindi

 hindilearn.in/tut/python/python-keywords-in-hindi

Python Keyword in Python

Source Code : Output :

```
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except',  
'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return',  
'try', 'while', 'with', 'yield']
```

<u>and</u>	<u>as</u>	<u>assert</u>	<u>break</u>
<u>class</u>	<u>continue</u>	<u>def</u>	<u>del</u>
<u>elif</u>	<u>else</u>	<u>except</u>	<u>nonlocal</u>
<u>False</u>	<u>finally</u>	<u>for</u>	<u>from</u>
<u>global</u>	<u>if</u>	<u>import</u>	<u>in</u>
<u>is</u>	<u>lambda</u>	<u>None</u>	<u>not</u>
<u>return</u>	<u>True</u>	<u>try</u>	<u>while</u>
<u>with</u>	<u>yield</u>		

'and'(Logical AND) Keyword in Python

'and' keyword का इस्तेमाल जब दिए गए दो या दो से ज्यादा operands true होते हैं तब true return किया जाता है |

0 = True

1 = False

x	y	x and y
0	0	0
0	1	1
1	0	1

1	1	1
---	---	---

Truth Table of 'and'

Source Code : Output :

False
True
False

'as' Keyword in Python

'as' keyword का इस्तेमाल module को कोई नया या उपनाम(alias) नाम देकर import किया जाता है |

Source Code :

```
import keyword as pyKey
print(pyKey.kwlist)
```

Output :

['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

'assert' Keyword in Python

Python Program में debugging के लिए 'assert' keyword का इस्तेमाल किया जाता है |

'assert' keyword के साथ जब condition true होता है तब कुछ नहीं होता और जब condition false होता है तब 'AssertionError' occur होता है |

Source Code :

```
assert 4==4 #No Exception
assert 4==5 #AssertionError Exception Occured
```

Output :

```
assert 4==5 #AssertionError Exception Occured
AssertionError
```

'break' Keyword in Python

Source Code :

```
nums = [1, 2, 3, 4, 5]
for n in nums :
    print(n)
    if(n == 4):
        break
```

Output :

```
1
2
3
4
```

'class' Keyword in Python

User द्वारा classes को define करने के लिए 'class' keyword का इस्तेमाल किया जाता है |

classes में कुछ attributes और कुछ methods होते हैं |

Classes 'OOP' में सबसे महत्वपूर्ण हिस्सा हैं |

Source Code :

```
class myClass:
    def func():
        print("Hello")

myClass.func()
```

Output :

```
Hello
```

'continue' Keyword in Python

'continue' keyword का इस्तेमाल iterate हो रहे कुछ statements को skip करता है |

'continue' का उपयोग iterate न हो रहे Loop को iterate करने के लिए किया जाता है |

Source Code : Output :

value of a is 0
value of a is 1
value of a is 2
value of a is 3
value of a is 4
skipped value of a is 5
value of a is 6
value of a is 7
value of a is 8
value of a is 9

'def' Keyword in Python

'def' keyword का इस्तेमाल user-defined function बनाने के लिए किया जाता है |

Source Code :

```
def func(a):  
    print("Value of a is", a)  
  
func(5)
```

Output :

Value of a is 5

'del' Keyword in Python

Python 'del' keyword का मतलब delete होता है |

'del' keyword का इस्तेमाल list, tuple, dictionary या किसी और collection से element को delete करने के लिए इस्तेमाल किया जाता है |

Source Code : Output :

[1, 4, 5, 7]

'del' keyword के बजाय remove() function का भी इस्तेमाल किया जाता है |

Source Code : Output :

[1, 4, 5, 7]

'elif' Keyword in Python

'elif' keyword का इस्तेमाल if और else keyword के साथ किया जाता है | एक से ज्यादा conditions को check करने के लिए 'elif' का इस्तेमाल किया जाता है |

Source Code : Output :

a is greater than 10

'else' Keyword in Python

'else' keyword का इस्तेमाल if, elif या सिर्फ if के साथ किया जाता है | अगर if या if और elif की condition false होती है तो else का statement execute होता है |

Source Code : Output :

Number is odd.

'except' Keyword in Python

'except' keyword का इस्तेमाल exception/error handling के लिए किया जाता है | try clause के साथ except को इस्तेमाल किया जाता है |

Example में try block में जो भी exception occur होगा उसका तुरंत Block execute हो जाएगा |

Example 1

Source Code :

```
try:
    a
except ZeroDivisionError:
    print("Divided by zero Error")
except NameError:
    print("a is not found")
```

Output :

a is not found

Example 2

Source Code :

```
try:
    4/0
except ZeroDivisionError:
    print("Divided by zero Error")
except NameError:
    print("a is not found")
```

Output :

Divided by zero Error

'nonlocal' Keyword in Python

'nonlocal' keyword का इस्तेमाल nested function के लिए किया जाता है | 'nonlocal' variables ये global भी नहीं होते और global भी नहीं होते हैं | अगर inner function में उनको लिया जाता है तो outer functions में उनकी values change नहीं होती हैं |

Without nonlocal

Source Code : Output :

```
subFunc : 10
mainFunc : 5
global : 2
```

With nonlocal

Source Code :

```
var = 2
def mainFunc():
    var = 5
    def subFunc():
        nonlocal var
        var = 10
        print("subFunc : ", var)

    subFunc()
    print("mainFunc : ", var)

mainFunc()
print("global : ", var)
```

Output :

```
subFunc : 10
mainFunc : 10
global : 2
```

'False' Keyword in Python

'False' keyword boolean false का वर्णन करता है | अगर दी गयी condition या statement गलत होता है तो False; return किया जाता है | False '0' के बराबर होता है |

Source Code : Output :

```
True  
False
```

'finally' Keyword in Python

'finally' keyword का इस्तेमाल 'try-except' या 'try' block के साथ किया जाता है |

exception/error हो या ना हो finally block execute होता है |

Source Code : Output :

```
NameError Exception occurred  
Always executed
```

'for' Keyword in Python

'for' keyword का इस्तेमाल looping के लिए किया जाता है | Collection के elements को iterate करने के लिए 'for' Loop का इस्तेमाल किया जाता है |

Source Code :

```
list = ["Rakesh", "Ramesh", "Suresh"]  
for n in list:  
    print(n)
```

Output :

```
Rakesh  
Ramesh  
Suresh
```

'from' Keyword in Python

'from' keyword का इस्तेमाल 'import' keyword के साथ किया जाता है | कुछ विशिष्ट function या attribute को import करने के लिए 'from' का इस्तेमाल किया जाता है |

अगर सिर्फ 'Module' को import किया जाता है और उसमे से func1() को लेना पड़ता है तो 'Module.func1()' लिखना पड़ता है |

Module.py

Source Code : Output :

sample.py

Source Code :

```
from Module import func1
func1()
```

Output :

I am in func1

'global' Keyword in Python

'global' keyword का इस्तेमाल function के variable को module में global बनाने के लिए किया जाता है |

Difference between Local and Global Variable in function

test1.py

Source Code : Output :

Local Variable

```
print(string)
```

NameError: name 'string' is not defined

test2.py

Source Code : Output :

Local Variable

Local Variable

'if' Keyword in Python

'if' keyword का इस्तेमाल if statement, if_else statement और if_elif_else statement में किया जाता है |

Source Code :

```
var = 0

if(var == 0):
    print("var is equal to 0")
```


Output :

```
var is equal to 0
```

'import' Keyword in Python

'import' keyword का इस्तेमाल module को current program पर import करने के लिए किया जाता है | Example पर math module को import किया गया है |

Source Code : Output :

```
4.0
```

'in' Keyword in Python

'in' keyword का इस्तेमाल collections(list, tuple, string, dictionary) में element है या नहीं ये check करके boolean value return करता है | for Loop में भी 'in' keyword का होना अनिवार्य होता है |

Source Code : Output :

```
True  
False
```

'is' Keyword in Python

'is' keyword का इस्तेमाल object equality के लिए किया जाता है | ये keyword '==' operator के थोड़ा-बहुत समान होता है | अगर दो object equal होते हैं तो 'True' return होता है और equal नहीं होते हैं तो 'False' return होता है |

Source Code : Output :

```
True  
False  
True  
True  
False  
True
```

'lambda' Keyword in Python

'lambda' keyword से anonymous(without name) function को create किया जाता है | name वाले function में 'def' keyword का इस्तेमाल किया जाता है और anonymous function में 'def' के बजाय 'lambda' keyword का इस्तेमाल किया जाता है | lambda function के लिए return statement नहीं होता है | function के arguments list को parenthesis() में close नहीं किया जाता है | anonymous function होने के कारण इनको किसी variable पर assign किया जाता है और उस variable के जरिये call किया जाता है |

निचे दिए गए दोनों Example का मतलब एक ही है |

Normal Function

Source Code : Output :

25

Anonymous Function

Source Code : Output :

25

'None' Keyword in Python

Python में 'None' keyword का इस्तेमाल किया जाता है लेकिन दूसरे Programming languages में 'null' keyword का इस्तेमाल किया जाता है | 'None' का मतलब कोई value नहीं होती | अगर function कोई value return नहीं करता है तो 'None' return करता है |

Source Code : Output :

```
False
False
True
<class 'NoneType'>
```

'not'(Logical NOT) Keyword in Python

'not' keyword का इस्तेमाल जब दिए गए Operand को true से false या false से true में convert करता है |

0 = True

1 = False

x	not x
0	1
1	0

Truth Table of

'and'

Source Code : Output :

False
True

'or'(Logical OR) Keyword in Python

'or' keyword का इस्तेमाल जब दिए गए operands में से एक भी operand true होता है तब true return किया जाता है |

0 = True
1 = False

x	y	x or y
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table of 'or'

Source Code : Output :

True
True
True
False

'pass' Keyword in Python

'pass' keyword का इस्तेमाल Loops, functions और classes को रिक्त रखा है | ये एक null statement/block होता है | 'pass' का इस्तेमाल जब किया जाता है तब ये कुछ नहीं करता है |

pass statement का इस्तेमाल एक ही line पर या separate line पर भी किया जाता है |

Source Code :

```
def func(): pass

while 4 < 5:
    pass
```

Output :

'raise' Keyword in Python

'raise' keyword का इस्तेमाल Exception Handling के लिए किया जाता है | raise keyword से custom exception/error का निर्माण किया जाता है | ज्यादातर programming languages में 'throw' keyword का इस्तेमाल किया जाता है लेकिन Python में 'raise' का इस्तेमाल किया जाता है |

Source Code :

```
a = 12
b = 10
if a > b:
    raise ValueError("a should be smaller than b")
```

Output :

Traceback (most recent call last):

```
raise ValueError('a should be smaller than b')
ValueError: a should be smaller than b
```

'return' Keyword in Python

'return' keyword का इस्तेमाल function के आखिरी में किया जाता है | function को अगर कोई value return नहीं की जाती है तो null/None return किया जाता है |

Source Code : Output :

25

'True' Keyword in Python

'True' keyword ये एक Boolean value है | Logical या Comparison operations में boolean value(True, False) return किया जाता है | True की value '1' और False की value '0' होती है |

Source Code : Output :

True
True
True
2

'try' Keyword in Python

'try' keyword का इस्तेमाल exception handling में किया जाता है | जिस code exception/error check करना है उस code को try Block पर लिखा जाता है |

Source Code :

```
try:
    a
except NameError:
    print("Value of a is not defined")
```

Output :

Value of a is not defined

'while' Keyword in Python

'while' keyword का इस्तेमाल looping के लिए किया जाता है | जब तक while condition true होती है तब तक statement iterate होता रहता है और अगर condition false या loop break किया जाता है तब loop का iteration बंद हो जाता है |

Source Code : Output :

Value of a is 0
Value of a is 1
Value of a is 2
Value of a is 3
Value of a is 4
Value of a is 5
Value of a is 6
Value of a is 7
Value of a is 8
Value of a is 9

'with' Keyword in Python

'with' keyword का इस्तेमाल file handling के लिए किया जाता है | जब file के लिए 'with_as' statement का इस्तेमाल किया जाता है तब file को close करने की जरूरत नहीं पड़ती है |

निचे दिए गए दोनों examples का मतलब एक ही है |

Without 'with'

Source Code :

```
file = open("textfile.txt", "w")
file.write("Hello World")
file.close()
```

textfile.txt

Hello World

With 'with'

Source Code :

```
with open("textfile.txt", "w") as file :
    file.write("Hello World")
```

textfile.txt

Hello World

'yield' Keyword in Python

'yield' keyword का इस्तेमाल function के लिए return statement जैसे किया जाता है | 'yield' keyword; generator को return करता है | return statement; function के आखिर में function terminate करता है लेकिन 'yield' statement; function को suspend करके वहा से आगे वापस शुरू होता है | Normal function जहा पर खत्म होता है वहा पर वापस नहीं आता है |

Source Code :

```
def func_name():
    yield 5
    yield 10
    yield 15

for y in func_name():
    print(y)
```

Output :

5
10
15

next() function का इस्तेमाल 'yield' के लिए किया जाता है | next() function generator object को लेकर अगली value return करता है |

Source Code : Output :

```
5  
10  
15
```