Variables in Hindi - Python in Hindi

What is Variable ?

Variable जब create किया जाता है तब interpreter द्वारा value को store करने के लिए memory location आरक्षित की जाती है |

Variable पर कोई भी data type की value store की जा सकती है | जैसे कि, Number, string, list, tuple, dictionary

Assigning Value to Variable

Python में declaration की जरुरत नहीं होती है | जब variable पर value assign होती है तब automatically declaration होता है |

declaration न होने के कारण Python में variable की default value नहीं होती है |

For Example,

```
a = 5 #Number
b = "Hello" #string
c = [2, 5, 9] #list
print(a, b, c)
```

Output :

```
5 Hello [2, 5, 9]
```

Changing Variable's Value

Python में variable की value change या re-assign की जा सकती है |

Source Code :

```
a = 5
print(a)
a = "Hello"
print(a)
a = [4, 5, 8]
print(a)
```

Output :

```
5
Hello
[4, 5, 8]
```

Assigning Single Value to Multiple Variables

Python में एक ही value एक से ज्यादा variables पर assign की जा सकती है |

Source Code :

```
a = b = c = d = "Hello"
print(a)
print(b)
print(c)
print(d)
```

Output :

```
Hello
Hello
Hello
Hello
```

Assigning Value to Variable according to order

Python में क्रमनुसार variable पर value store की जाती है |

Example पर एक ही memory location multiple variables और उनकी values assign की गयी है |

Source Code :

```
a, b, c = 1, 'H', [1, 2]
print(a)
print(b)
print(c)
```

Output :

```
1
H
[1, 2]
```

Variables Concatenation

Python में एक ही data types के variables concatenate किय जा सकते है |

Example पर str() function का इस्तेमाल object को integer से string में convert करने के लिए किया गया है |

Source Code :

```
a = 1
b = 2
print(a + b)
print(str(a) + str(b))
c = "Hello"
print(str(a) + c)
```

Output :

```
3
12
1Hello
```

Types of Variables

Python में variable के दो प्रकार है |

1. Local Variables
2. Global Variables

1. Local Variables

Local Variables; functions के अन्दर होते है | उनकी visibility सिर्फ function के अन्दर होती है, जब वो function के बाहर आते है तब destroy हो जाते है |

Source Code :

```
def func():
 a = 5 #local variable
 print(a)
func()
print(a)
```

Output :

```
5
Traceback (most recent call last):
 print(a)
NameError: name 'a' is not defined
```

2. Global Variables

Global Variables; function के बाहर होते है | उनकी visibility function के अन्दर और बाहर होती है |

उनका scope पूरे program पर होता है |

Source Code :

```
a = 10 #global variable
def func():
 print(a)
func()
print(a)
```

Output :

```
10
10
```

Example पर local और global ये दोनों variables declared किये गए है | function के बाहर का variable lobal है और अन्दर का variable local है | global variable का scope function के अन्दर और बाहर होता है लेकिन function के अन्दर अलग से variable declaration होने के कारण func() call करते ही variable की value change हो जाती है |

Source Code :

```
a = 10 #global variable
def func():
 a = 5 #local variable
 print(a)
func() #print local
print(a) #print global
```

Output :

```
5
10
```

With 'global' and Without 'global' Keyword

function में variable के लिए 'global' keyword का भी इस्तेमाल किया जाता है |

Without global

Example पर 'a' variable के declaration से पहले ही 'a' variable को print किया गया है, इसके कारण 'UnboundLocalError' ये exception occur हुआ है |

Source Code :

```
def func():
 print(a)
 a = "local"
 print(a)
a = "global"
func()
print (a)
```

Output :

```
 in func
 print(a)
UnboundLocalError: local variable 'a' referenced before assignment
```

With global

Example पर global keyword का इस्तेमाल शुरुआत में ही किया गया है और उसके बाद variable a को print किया गया है | program में जहा पर भी 'a' नामक global variable होगा वो func() call करते ही पहले print हो जायेगा |

Source Code :

```
def func():
 global a
 print(a) #print global
 a = "local"
 print(a) #print local
a = "global"
func()
print(a) #print local
```

Output :

```
global
local
local
```

No-Local or No-Global Variable

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

Without nonlocal
Source Code :

```
var = 2
def outer():
 var = 5
 def inner():
 var = 10
 print("inner : ", var)
 inner()
 print("outer : ", var)
outer()
print("global : ", var)
```

Output :

```
inner : 10
outer : 5
global : 2
```

With nonlocal

Source Code :

```
var = 2
def outer():
 var = 5
 def inner():
 nonlocal var
 var = 10
 print("inner : ", var)
 inner()
 print("outer : ", var)
outer()
print("global : ", var)
```

Output :

```
inner : 10
outer : 10
global : 2
```
