# Python Variables

In [1]:

```
n='v'
n
```

Out[1]:

```
'v'
```

In [2]:

```
id(n) # address of variable n in memory
```

Out[2]:

```
4374587792
```

In [4]:

```
# variables as case sensitive
n1=2
N1
```

```
---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
Cell In[4], line 2
      1 n1=2
----> 2 N1

NameError: name 'N1' is not defined
```

In [5]:

```
# varaible names can't start with numbers
8=n2
n2
```

```
  Cell In[5], line 1
    8=n2
    ^
SyntaxError: cannot assign to literal here. Maybe you meant '==' instead o
f '='?
```

In [6]:

```
# variable names can ends with numbers
n1=2
n1
```

Out[6]:

```
2
```

In [8]:

```
# variable names can't contain any special chars except _
n@n=10
n@n
```

```
  Cell In[8], line 1
    n@n=10
    ^
SyntaxError: cannot assign to expression here. Maybe you meant '==' instea
d of '='?
```

In [9]:

```
n_n=10
n_n
```

Out[9]:

```
10
```

In [10]:
```
_n1=10
_n1
```

Out[10]:
```
10
```

In [11]:
```
n1_=9
n1_
```

Out[11]:
```
9
```

In [12]:
```
# variable names could not be reserved keywords
import keyword
keyword.kwlist
```

Out[12]:
```
['False',
 'None',
 'True',
 'and',
 'as',
 'assert',
 'async',
 'await',
 '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']
```

In [13]:
```
if=2
if
```

```
  Cell In[13], line 1
    if=2
      ^
SyntaxError: invalid syntax
```

In [14]:

```python
If=2
If
```

Out[14]:

```
2
```

# Work with print statement

In [18]:

```python
age=31
price=12.50
name="kalyani"
isActive=True
print(age,price,name,isActive)
```

```
31 12.5 kalyani True
```

In [20]:

```python
# changing varaible value
x=10
print(x)
x=20
print(x)
```

```
10
20
```

In [23]:

```python
# strings cancatenating
fName="kalyani"
lName="a"
print(fName+" "+lName)
```

```
kalyani a
```

In [27]:

```python
# Print result with string
a=10
b=20
print("addition of",a,b,"is",a+b)
```

```
addition of 10 20 is 30
```

In [25]:

```python
# Print format method
a=10
b=20
print("addition of {},{} is {}".format(a,b,a+b))
```

```
addition of 10,20 is 30
```

In [26]:

```python
# fstring method-more short format
a=10
b=20
print(f"addition of {a},{b} is {a+b}")
```

```
addition of 10,20 is 30
```

In [30]:

```python
# print with end and separator
print("I'm",end="#")
print("kalyani")

print("I'm","kalyani",sep="#")
print(1,2,end=" ")
print(3,".",sep="")
```

```
I'm#kalyani
I'm#kalyani
1 2 3.
```

In [32]:
```
# print with quote in quotes
print("isn't it?")
print('isn"t it?')
```

```
isn't it?
isn"t it?
```

In [36]:
```
# print with quote in quotes with escape \
print("isn\"t it?")
print('isn\'t it?')
print("'isn\"t' they said?")
```

```
isn"t it?
isn't it?
'isn"t' they said?
```

In [40]:
```
# print with special chars
s='first line\nsecondline'
s
```

Out[40]:
```
'first line\nsecondline'
```

In [41]:
```
print(s)
```

```
first line
secondline
```

In [43]:
```
# print with raw strings
location='C:\some\name'
print(location)
print(r'C:\some\name')
```

```
C:\some
ame
C:\some\name
```

In [44]:
```
# string cancatination
3*'un'+'num'
```

Out[44]:
```
'unununnum'
```

In [45]:
```
s1='abc'
3*'un'+s1
```

Out[45]:
```
'unununabc'
```

In [46]:
```
'py' 'thon'
```

Out[46]:
```
'python'
```

In [78]:
```
s1='py'
s1 'thon'
```

```
  Cell In[78], line 2
    s1 'thon'
       ^
SyntaxError: invalid syntax
```

In [49]:
```
# indexing and slicing of strings
name='kalyani'
name[0]
```

Out[49]:
```
'k'
```

In [50]:
```
name[1]
```

Out[50]:
```
'a'
```

In [51]:
```
name[6]
```

Out[51]:
```
'i'
```

In [52]:
```
name[-1]
```

Out[52]:
```
'i'
```

In [53]:
```
name[-2]
```

Out[53]:
```
'n'
```

In [54]:
```
name[-7]
```

Out[54]:
```
'k'
```

In [55]:
```
name[1:2]
```

Out[55]:
```
'a'
```

In [56]:
```
name[1:7]
```

Out[56]:
```
'alyani'
```

In [57]:
```
name[-1:-6]
```

Out[57]:
```
''
```

In [58]:
```
name[-6:-1]
```

Out[58]:
```
'alyan'
```

In [59]:
```
name[-5:-6]
```

Out[59]:
```
''
```

In [60]:
```
name[-6:-5]
```

Out[60]:
```
'a'
```

In [61]:
```
name[1:]
```

```
Out[61]:  'alyani'

In [62]:  name[:7]

Out[62]:  'kalyani'

In [63]:  name[:2]

Out[63]:  'ka'

In [64]:  name[:]

Out[64]:  'kalyani'

In [65]:  name[:2]+name[2:]

Out[65]:  'kalyani'

In [66]:  # index out of range with indexing
          name[7]
```

```
---------------------------------------------------------------------------
-
IndexError                                Traceback (most recent call las
t)
Cell In[66], line 2
      1 # index out of range error
----> 2 name[7]

IndexError: string index out of range
```

```
In [68]:  # index out of range with slicing
          name[1:30]

Out[68]:  'alyani'

In [69]:  name[:90]

Out[69]:  'kalyani'

In [70]:  name[20:]

Out[70]:  ''

In [73]:  # mutate strings
          name[2]='v'
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[73], line 2
      1 # mutate strings
----> 2 name[2]='v'

TypeError: 'str' object does not support item assignment
```

In [74]:
```
name[2:3]='v'
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[74], line 1
----> 1 name[2:3]='v'

TypeError: 'str' object does not support item assignment
```

In [76]:
```
len(name)
```

Out[76]:
```
7
```

# python list datatype

In [79]:
```
squares=[1,4,9,16,25]
squares
```

Out[79]:
```
[1, 4, 9, 16, 25]
```

In [82]:
```
# indexing
squares[0]
```

Out[82]:
```
1
```

In [83]:
```
squares[4]
```

Out[83]:
```
25
```

In [84]:
```
squares[5]
```

```
---------------------------------------------------------------------------
-
IndexError                                Traceback (most recent call las
t)
Cell In[84], line 1
----> 1 squares[5]

IndexError: list index out of range
```

In [85]:
```
squares[-2]
```

Out[85]:
```
16
```

In [86]:
```
squares[-1]
```

Out[86]:
```
25
```

In [87]:
```
squares[-3:-1]
```

Out[87]:
```
[9, 16]
```

In [88]:
```
squares[1:]
```

```
Out[88]:  [4, 9, 16, 25]
```

```
In [89]:  squares[:5]
```

```
Out[89]:  [1, 4, 9, 16, 25]
```

```
In [92]:  # cancate
          squares=squares+[36,49]
          squares
```

```
Out[92]:  [1, 4, 9, 16, 25, 36, 49]
```

```
In [94]:  # mutation
          squares[3]=17
          squares
```

```
Out[94]:  [1, 4, 9, 17, 25, 36, 49]
```

```
In [95]:  #append
          squares.append(64)
          squares
```

```
Out[95]:  [1, 4, 9, 17, 25, 36, 49, 64]
```

```
In [103…  # normal copy with lists
          rgb=["red","green","blue"]
          rgba=rgb
          rgba.append('alpha')
          print(rgb,rgba)

          # shallow copy
          rgb=["red","green","blue"]
          rgba=rgb[:]
          rgba.append('alpha')
          print(rgb,rgba)
```

```
['red', 'green', 'blue', 'alpha'] ['red', 'green', 'blue', 'alpha']
['red', 'green', 'blue'] ['red', 'green', 'blue', 'alpha']
```

```
In [112…  # assignment to slices is also possible with lists
          letters=['a','b','c','d','e']
          letters[0]="A"
          print(letters)
          letters[1:3]=['B','C']
          print(letters)
          letters[1:3]=['B']
          print(letters)
          letters[1:]=['B']
          print(letters)
          letters[:]=[]
          print(letters)
```

```
['A', 'b', 'c', 'd', 'e']
['A', 'B', 'C', 'd', 'e']
['A', 'B', 'd', 'e']
['A', 'B']
[]
```

```
In [113…  len(letters)
```

Out[113… 0

In [115…

```
# nested lists
a=['a','b','c']
n=[1,2]
x=[a,n]
x
```

Out[115…

```
[['a', 'b', 'c'], [1, 2]]
```

In [116…

```
x[0]
```

Out[116…

```
['a', 'b', 'c']
```

In [117…

```
x[0][2]
```

Out[117…

```
'c'
```

In [118…

```
# fibonacci series
a,b=0,1
while(a<1000):
    print(a,end=" ")
    a,b=b,a+b
```

```
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

In [ ]: