Numbers

```
In [1]: 1 + 1

Out[1]: 2

In [2]: 1 * 3

Out[2]: 3

In [3]: 1 / 2

Out[3]: 0.5

In [4]: 2 ** 4

Out[4]: 16

In [5]: 4 % 2

Out[5]: 0

In [6]: 5%2

Out[6]: 1

In [7]: (2+3)*(5+5)

Out[7]: 50
```

Variable Assignment

```
In [51]: name_of_var = 2

In [52]: x = 2
         y = 3

In [53]: z = x + y
         z

Out[53]: 5
```

Strings

```
In [54]:  'single quotes'

Out[54]:  'single quotes'

In [55]:  "single quotes"

Out[55]:  'single quotes'

In [56]:  " wrap lot's of other so-called 'quotes' "

Out[56]:  " wrap lot's of other so-called 'quotes' "

In [57]:  " wrap lot's of other 'so-called quotes'"

Out[57]:  " wrap lot's of other 'so-called quotes'"
```

Printing

```
In [58]:  x = 'hello'

In [59]:  x

Out[59]:  'hello'

In [60]:  print(x)

          hello

In [61]:  num = 12
          name = 'Sam'

In [62]:  print('My number is: {one}, and my name is: {two}'.format(one=name,two=num))

          My number is: Sam, and my name is: 12

In [63]:  print('My number is: {}, and my name is: {}'.format(name,num))

          My number is: Sam, and my name is: 12
```

Lists

```
In [64]:  [1,2,3]

Out[64]:  [1, 2, 3]
```

```
In [65]: ['hi',1,[2,3]]

Out[65]: ['hi', 1, [2, 3]]

In [66]: a_list = ['hi',1,[2,3]]

In [67]: a_list[0]

Out[67]: 'hi'

In [68]: a_list[1]

Out[68]: 1

In [69]: a_list[2]

Out[69]: [2, 3]

In [70]: a_list[2][0]

Out[70]: 2

In [71]: a_list[2][1]

Out[71]: 3

In [72]: a_list.append(4)

In [73]: a_list.append([5,6,7])

In [74]: a_list

Out[74]: ['hi', 1, [2, 3], 4, [5, 6, 7]]

In [75]: a_list[:1]

Out[75]: ['hi']

In [76]: a_list[1:]

Out[76]: [1, [2, 3], 4, [5, 6, 7]]
```

```
In [77]:  a_list[:0]

Out[77]:  []

In [78]:  a_list[3:]

Out[78]:  [4, [5, 6, 7]]

In [79]:  a_list[0] = 'hey'
          a_list

Out[79]:  ['hey', 1, [2, 3], 4, [5, 6, 7]]

In [80]:  nest = [1,2,3,[4,5,['target',99]]]

In [81]:  nest[3][2][1]

Out[81]:  99

In [82]:  nest[3][2][0]

Out[82]:  'target'
```

Dictionary

```
In [83]:  d = {'key1':'item1','key2':'item2'}
          d

Out[83]:  {'key1': 'item1', 'key2': 'item2'}

In [84]:  d['key2']

Out[84]:  'item2'

In [85]:  my_dict = {
              'hello':'world',
              'test':3,
              'foo':'bar'
          }

In [86]:  my_dict['hello']

Out[86]:  'world'
```

```
In [87]: my_dict['test']

Out[87]: 3

In [88]: my_dict['foo']

Out[88]: 'bar'
```

Tuples

```
In [89]: t = (1,2,3)

In [90]: t[0]

Out[90]: 1

In [91]: t[2]

Out[91]: 3

In [92]: t

Out[92]: (1, 2, 3)

In [93]: t[0] = "One"

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [93], in <cell line: 1>()
----> 1 t[0] = "One"

TypeError: 'tuple' object does not support item assignment

In [94]: t[0] = 0

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [94], in <cell line: 1>()
----> 1 t[0] = 0

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

Sets

```
In [95]: {1,2,3}

Out[95]: {1, 2, 3}
```

In [96]:
```
{1,2,3,3,4,4,4,5,6,7,7,7,7,7}
```

Out[96]:
```
{1, 2, 3, 4, 5, 6, 7}
```

Comparison Operators

In [97]:
```
1>2
```

Out[97]:
```
False
```

In [98]:
```
1<2
```

Out[98]:
```
True
```

In [99]:
```
1>=1
```

Out[99]:
```
True
```

In [100…
```
1<=1
```

Out[100…
```
True
```

In [101…
```
1==1
```

Out[101…
```
True
```

In [102…
```
'hi' == 'bye'
```

Out[102…
```
False
```

In [103…
```
'hi' == 'hi'
```

Out[103…
```
True
```

Logic Operators

In [104…
```
(1>2)and(2<3)
```

Out[104…
```
False
```

In [105…
```
(1>2)or(2<3)
```

Out[105…
```
True
```

if,else,else if

In [106…

```
if 1<2:
    print('yep!')
print('Print anyway')
```

```
yep!
Print anyway
```

In [107…

```
if 1>=2:
    print('yep!')
print('Print anyway')
```

```
Print anyway
```

In [108…

```
if 1>2:
    print('first')
else:
    print('second')
```

```
second
```

In [109…

```
if 1<2:
    print('first')
else:
    print('second')
```

```
first
```

In [110…

```
if 1==2:
    print('first')
elif 2==2:
    print('second')
else:
    print('last')
```

```
second
```

In [111…

```
if 1==2:
    print('first')
elif 2!=2:
    print('second')
else:
    print('last')
```

```
last
```

For loops

In [112…

```
seq = [1,2,3,4,5]
```

In [113…

```
for item in seq:
    print(item)
```

```
1
```

```
2
3
4
5
```

In [114…

```
for item in seq:
    if item <= 3:
        print("lesser or equal")
    else:
        print("Greater")
```

```
lesser or equal
lesser or equal
lesser or equal
Greater
Greater
```

In [115…

```
for item in seq:
    print(item ** 3)
```

```
1
8
27
64
125
```

In [116…

```
for item in seq:
    print(item + item)
```

```
2
4
6
8
10
```

While Loops

In [117…

```
i = 1
while i < 5:
    print('i is: {}'.format(i))
    i = i + 1
```

```
i is: 1
i is: 2
i is: 3
i is: 4
```

Range

In [118…

```
range(7)
```

Out[118…

```
range(0, 7)
```

In [119…

```
for i in range(7):
    print(i)
```

```
0
1
2
3
4
5
6
```

In [120…

```
list(range(7))
```

Out[120…

```
[0, 1, 2, 3, 4, 5, 6]
```

In [121…

```
set(range(7))
```

Out[121…

```
{0, 1, 2, 3, 4, 5, 6}
```

In [122…

```
tuple(range(7))
```

Out[122…

```
(0, 1, 2, 3, 4, 5, 6)
```

In [123…

```
for n in range(10,20):
    print(n)
```

```
10
11
12
13
14
15
16
17
18
19
```

List Comprehension

In [124…

```
for n in range(10,20):
    print(n)
```

```
10
11
12
13
14
15
16
17
18
19
```

In [125…

```
x = [1,2,3,4]
```

In [126…

```
out = []
for item in x:
    out.append(item ** 2)
print(out)
```

```
[1, 4, 9, 16]
```

In [127…

```
[item**2 for item in x]
```

Out[127…

```
[1, 4, 9, 16]
```

In [128…

```
x3 = [item**3 for item in x]
x3
```

Out[128…

```
[1, 8, 27, 64]
```

Functions

In [129…

```
def square_num(input=0):
    """
    Default input is 0
    This function squares the input value provided
    """
    result = input ** 2
    print(result)
```

In [130…

```
square_num
```

Out[130…

```
<function __main__.square_num(input=0)>
```

In [131…

```
square_num()
```

```
0
```

In [132…

```
square_num(input = 5)
```

```
25
```

In [133…

```
square_num(5)
```

```
25
```

In [134…

```
def cube(x):
    return x**3
```

In [135…

```
cubes = cube(3)
cubes
```

Out[135…

```
27
```

In [136…

```
cubes_2=[cube(x) for x in range(1,6)]
```

In [137…

```
cubes_2
```

Out[137…

```
[1, 8, 27, 64, 125]
```

In [138…

```
def hello_person(name=None):
    if name == None:
        print("Hello!")
    else:
        print("Hello " + name + "!")
```

In [139…

```
hello_person()
```

```
Hello!
```

In [140…

```
hello_person("Yasin")
```

```
Hello Yasin!
```

Lambda Expressions

In [141…

```
var_lambda = lambda test:test*3
```

In [142…

```
print(var_lambda(3))
```

```
9
```

In [143…

```
[var_lambda(i) for i in range(9)]
```

Out[143…

```
[0, 3, 6, 9, 12, 15, 18, 21, 24]
```

Map and filter

In [144…

```
seq = [1,2,3,4]
```

In [145…

```
seq
```

Out[145…

```
[1, 2, 3, 4]
```

```
In [146…  map(var_lambda,seq)

Out[146…  <map at 0x7faf816adca0>

In [147…  list(map(lambda lambda_exp:2**lambda_exp,seq))

Out[147…  [2, 4, 8, 16]

In [148…  set(map(lambda lambda_exp:2**lambda_exp,seq))

Out[148…  {2, 4, 8, 16}

In [149…  tuple(map(lambda lambda_exp:2**lambda_exp,seq))

Out[149…  (2, 4, 8, 16)

In [150…  lambda_e = lambda lambda_exp:2**lambda_exp,seq
          li = list(map(lambda_e,seq))
          print(li)
          # Works on
          # https://www.w3schools.com/python/trypython.asp?filename=demo_lambda
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [150], in <cell line: 2>()
      1 lambda_e = lambda lambda_exp:2**lambda_exp,seq
----> 2 li = list(map(lambda_e,seq))
      3 print(li)

TypeError: 'tuple' object is not callable
```

```
In [151…  filter(lambda item:item%2==0,seq)

Out[151…  <filter at 0x7faf816ad850>

In [152…  list(filter(lambda item:item%2==0,seq)) # even numbers

Out[152…  [2, 4]

In [153…  list(filter(lambda item:item%2>=1,seq)) # odd numbers

Out[153…  [1, 3]
```

Methods

```
In [154…  st = 'Hello My Name Is Yasin'
In [155…  st

Out[155… 'Hello My Name Is Yasin'

In [156…  st.lower()

Out[156… 'hello my name is yasin'

In [157…  st.upper()

Out[157… 'HELLO MY NAME IS YASIN'

In [158…  st.split()

Out[158… ['Hello', 'My', 'Name', 'Is', 'Yasin']

In [159…  tweet = "Go Galatasaray, #Sports"

In [160…  tweet.split('#')

Out[160… ['Go Galatasaray, ', 'Sports']

In [161…  tweet.split(',')

Out[161… ['Go Galatasaray', ' #Sports']

In [162…  tweet.split('#')[0]

Out[162… 'Go Galatasaray, '

In [163…  tweet.split('#')[1]

Out[163… 'Sports'

In [164…  my_dict

Out[164… {'hello': 'world', 'test': 3, 'foo': 'bar'}

In [165…  my_dict.keys()

Out[165… dict_keys(['hello', 'test', 'foo'])
```

```
In [166…  my_dict.values()

Out[166…  dict_values(['world', 3, 'bar'])

In [167…  my_dict.items()

Out[167…  dict_items([('hello', 'world'), ('test', 3), ('foo', 'bar')])

In [168…  lst = [1,2,3,4,5,6,7,8,9,10]

In [169…  lst.pop()

Out[169…  10

In [170…  lst

Out[170…  [1, 2, 3, 4, 5, 6, 7, 8, 9]

In [171…  'x' in lst

Out[171…  False

In [172…  8 in lst

Out[172…  True

In [173…  'x' in [1, 2, 3, 4, 5, 6, 7, 8, 9]

Out[173…  False

In [174…  3 in [1, 2, 3, 4, 5, 6, 7, 8, 9]

Out[174…  True
```