Table of contents

---

1. Simultaneous Assignment

(go to top)

In [1]:
```
x = 2
y = 4
```

In [2]:
```
x = y
y = x
print(x,y)
```

```
4 4
```

In [3]:
```
x = 2
y = 4
```

In [4]:
```
x,y = y,x
print(x,y)
```

```
4 2
```

In [5]:
```
lst = [1, 2, 3]
lst[0] = lst [2]
lst
# here the value of list[0] is replaced
```

Out[5]:
```
[3, 2, 3]
```

In [6]:
```
lst = [1, 2, 3]
lst[0], lst[2] = lst [2], lst[0]
lst
# here the values are swapped
```

Out[6]:
```
[3, 2, 1]
```

---

2. Imports

(go to top)

In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import statsmodels as sm
import datetime as dt
from random import *
from math import * # no need to use math.sqrt() now we can just use sq
```

3. print new line \n, end='' and sep=''

(go to top)

In [6]:
```
print('This program generates a username for you.')
print('see there is a blank line above me')
```

```
This program generates a username for you.
see there is a blank line above me
```

In [7]:
```
print('This program generates a username for you. \n')
print('see now there is a blank line above me')
```

```
This program generates a username for you.

see now there is a blank line above me
```

In [2]:
```
nums = [1, 2, 3, 4, 5]
print (* iter(nums), sep='')
```

```
12345
```

In [6]:
```
# SEP NOT APPLICABLE

for i in nums:
    print(i, sep='')
```

```
1
2
3
4
5
```

In [9]:
```
# END NOT APPLICABLE

nums = [1, 2, 3, 4, 5]
print (* iter(nums), end='')
```

```
1 2 3 4 5
```

In [10]:
```
for i in nums:
    print(i, end='')
```

```
12345
```

4. map() to input.split()

(go to top)

In [8]:
```
#arr = map(int, input().split())

# do not enter decimal numbers
# python will not in a string representations of a float
# if you use int, input cannot be anything but string rep of an int
# to be safe use float, and int it later.

# enter floats seprated by space
arr = map(float, input().split())
```

```
3.5 6.2 3.5
```

In [9]:
```
a = list(arr)
print(a)
print(a[0])
type(a[0])
```

```
[3.5, 6.2, 3.5]
3.5
```

Out[9]:
```
float
```

5. For Loop over input

(go to top)

```
for i in range(int(input())):
        name = input()
        score = float(input())
```

---

6. Enumerate

(go to top)

In [10]:
```
avengers = ["hawkeye", "iron man", "thor", "quicksilver"]
```

In [11]:
```
e = enumerate(avengers)
d = list(e)
print(d)
```

```
[(0, 'hawkeye'), (1, 'iron man'), (2, 'thor'), (3, 'quicksilver')]
```

In [12]:
```
d[1][1]
```

Out[12]:
```
'iron man'
```

In [13]:
```
min(d)
```

Out[13]:
```
(0, 'hawkeye')
```

In [14]:
```
e = enumerate(avengers, start = 10)
print(list(e))
```

```
[(10, 'hawkeye'), (11, 'iron man'), (12, 'thor'), (13, 'quicksilver')
]
```

In [15]:
```
avengers = [2, 3, 4, 5]
```

In [16]:
```
e = enumerate(avengers)
d = list(e)
print(d)
```

```
[(0, 2), (1, 3), (2, 4), (3, 5)]
```

In [17]:
```
d[1][1]
```

Out[17]: 3

In [18]:
```
min(d)
```

Out[18]: (0, 2)

In [19]:
```
min_num = list(map(lambda num: min(d), d))
```

7. if name = main

(go to top)

```
def main():
    some function

if__name__ == '__main__':
    #run main
    main()


OR

if__name__ == '__main__':

    some algo
```

---

8. set()

(go to top)

- will remove duplicates and create dictinary in random order

In [20]:
```
lst = ['apple', 'banana', 'apple', 'orange']
lst_2 = set(lst)
```

In [21]:
```
lst_2
```

Out[21]:
```
{'apple', 'banana', 'orange'}
```

In [ ]:
```

```

---

9. Find Runnerup

- Winner is largest score, runner up is second largest score

(go to top)

In [23]:
```
# assuming inputs are integers only, for float use map(float, input.sp
# collect list of scores
print('Enter scores sperated by space')
arr = map(int, input().split())

# intitialise list of scores to scoreList
scoreList = list(arr)

# set the first score to be the bigges number
# make the runner up
winner = scoreList[0]
runnerUp = winner


# if the next number is greater than the winner make the next number t
# former winner

# if the next number is less than the current winner but bigger than t

# if the runner up and the winning number are the same, and the next n
# mae

for i in scoreList:

    if i > winner:
        runnerUp = winner;
        winner = i
        #print('that ran')
    elif i > runnerUp and i < winner:
        runnerUp = i
        #print('this ran')
    elif runnerUp == winner and i < winner:
        runnerUp = i


print('\n', winner, runnerUp)
```

```
Enter scores sperated by space
2 3 5 4 1 2 5 8 1 82 3 5 4 7

 82 8
```

10. *text to rest of split.()

(go to top)

In [24]:
```
text = ' hi 24 35 36'
```

In [25]:
```
words, *nums = text.split()
```

In [26]:
```
print(words)
print(nums)
```

```
hi
['24', '35', '36']
```

11. List to String

(go to top)

In [27]:
```
lst = ['s', 't', 'h', 'y', 'u', 'i']
string(list)
```

```
---------------------------------------------------------------------
------
NameError                                 Traceback (most recent call
last)
<ipython-input-27-4a009018bd21> in <module>
      1 list = ['s', 't', 'h', 'y', 'u', 'i']
----> 2 string(list)

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

In [29]:
```
"".join(lst)
```

Out[29]:
```
'sthyui'
```

12. use eval to run concatennated string command

(go to top)

In [30]:
```
lst = []
```

In [31]:
```
sample_input = ['insert 0 5', 'insert 1 10', 'insert 0 6', 'print', 'r
                'append 1', 'sort', 'print', 'pop', 'reverse', 'print'
```

In [32]:
```
for j in sample_input:
    inpt = j.split()

    #inpt format is ['insert', '0', '5'] or ['print'] or ['remove', '6
    command, arguments = inpt[0], inpt[1:]

    if command == 'print':
        print(lst)
    else:
        # run = 'lst.' + command + '(' + 'inpt[1]' + 'inpt[2]' + ')'
        # this will cause errors cos it will say list out of range whe
        # using inpt[1:] wont return errors just an empy list
        # ald not the best approac because what if the length of inpt
        # given lists the max argument number is probably about 3 thou
        # run = 'lst.' + command + '(' + 'inpt[1:2]' + 'inpt[2:3]' + '

        #-------LOOK HERE---------------
        run = 'lst.' + command + '(' + ','.join(arguments) + ')'
        eval(run)
```

```
[6, 5, 10]
[1, 5, 9, 10]
[9, 5, 1]
```

- This is the long version

In [1]:
```
#N = int(input())
lst = []

sample_input = ['insert 0 5', 'insert 1 10', 'insert 0 6', 'print', 'r
                'append 1', 'sort', 'print', 'pop', 'reverse', 'print'

for j in sample_input:
    command, *rest = j.split()
    if command == 'insert':
        rest_val = list(map(int, rest))
        lst.insert(rest_val[0], rest_val[1])
    if command == 'append':
        lst.append(int(rest[0]))
    if command == 'remove':
        lst.remove(int(rest[0]))
    if command == 'sort':
        lst.sort()
    if command == 'pop':
        lst.pop(-1)
    if command == 'reverse':
        lst.reverse()
    if command == 'print':
        print(lst)
```

```
[6, 5, 10]
[1, 5, 9, 10]
[9, 5, 1]
```

13. Quick repeating list

(go to top)

In [3]:
```
dice = [0]*5
dice
```

Out[3]:
```
[0, 0, 0, 0, 0]
```

In [ ]:
```

```

14. Reverse list slicing

(go to top)

In [1]:
```
lst = ['1', '2', '3', '4', '5']
```

In [57]:
```
lst[5::-1]
```

Out[57]:
```
['5', '4', '3', '2', '1']
```

In [56]:
```
lst[::-1]
```

Out[56]:
```
['5', '4', '3', '2', '1']
```

15. any()

(go to top)

In [2]:
```
lst = [1, 2, 3, 4, False]
```

In [4]:
```
# is there any item in lst that is True
any(lst)
```

Out[4]:
```
True
```

In [ ]:
```

```

16. type.method(variable of said type)

(go to top)

In [5]:
```
'a'.isupper()
```

Out[5]: False

In [7]:
```
type('a').isupper('a')
```

Out[7]: False

In [8]:
```
str.isupper('a')
```

Out[8]: False

In [10]:
```
type('a')
```

Out[10]: str

In [13]:
```
s = 'yyy'

for method in [str.isalnum, str.isalpha, str.isdigit, str.islower, str
    print(any(method(c) for c in s))
```

```
True
True
False
True
False
```

17. Title

(go to top)

18. Title

(go to top)

19. Title

(go to top)

20. Title

(go to top)

4. Title

(go to top)

5. Title

(go to top)

6. Title

(go to top)

7. Title

(go to top)

8. Title

(go to top)

9. Title

(go to top)

10. Title

(go to top)

11. Title

(go to top)

12. Title

(go to top)

13. Title

(go to top)

14. Title

(go to top)

15. Title

(go to top)

16. Title

(go to top)

17. Title

(go to top)

18. Title

(go to top)

19. Title

(go to top)

20. Title

(go to top)