Python-tutorial 2

- m = list("MADHU") (∵ list is func·)
  output: m → ['M', 'A', 'D', 'H', 'U'] internal iteration
  m = list ("MADHU", "JAIN")
  output: m → error.
  m = list (["MADHU", "JAIN"])
  output: m → ['MADHU', 'JAIN']
- string indexing:
  S = "MADHU" (∵ upper bound is excluded)
  S[0] = 'M' , S[0:3] = 'MAD', S[3:1:-1] = 'HD'
  S[3:1] = '' (∵ by default step-size = +1).
- Append: (append obj. to the end of list)
  l = [ ]
  l.append('sudh') → l = ['sudh']
  l.append ('abc') → l = ['sudh', 'abc']
  l.append(["xy3", 1, 2, 9]) → l = ['sudh', 'abc', ['xy3', 1, 2, 9]].
- If you wish to append data in b/w the list as per your choice use, insert function.

l.insert (1, ["I", "am"])

$\uparrow$ index at which you want to insert.

→ l = ['sudh', ['I', 'am'], 'abc', ['xy3', 1, 2, 9]]

→ l[1] = ['I', 'am']
 l[1][0] = 'I'. & l[1][1] = 'am'.

```
list(l[1](1)) = ['a', 'm']
```

• Addition :- (only when we have same data type)

```
l1 = [2, 5, 6, 7]
l2 = ["MAD", 8]
l1 + l2 = [2, 5, 6, 7, 'MAD', 8].
```

```
k = "MAD"
l1 + k = error (l1 is list but k is string)
```

```
l = ["MADHU", "JAIN"]
l[1][: : -1] = NIAJ
```

```
l = []
l.append('mad') → ['mad']
l.append('jain') → ['mad, 'jain']
l.insert(-1, ["Abc", "d"])
        → ['mad', ['Abc', 'd'], jain]
                     (-2)
```

```
l = [['abc', 'def'], 'efg', 'ijk', ['xy3', uvw']]
m = []
for i in l:
    print(i)
    m.append(i[: : -1])
↳ [['def', 'abc'], gfe', 'kji', ['uvw', 'xy3']]
```

```
• l[]
  for i in [1,2,3,"sudh",["def","xy3,"abc"]]:
      if type(i)==list:
          for j in i[: : -1]:
              l.append(j[: : -1])
  l
→  ['cba','3yx','fed']
```

```
• [ for i in "MAD":
        print(i)          → [M      (by default
                              A       vertical)
                              D]
```

```
  [ for i in "MAD":
        print(i, end='\n or \t')  → [M    (or ~~[illegible]~~
                                      A
                                      D]
                  by default          [M A D]
```

```
  [ for i in "~~A~~MAD":
        print(i, end='')  → [MAD]
```

```
  [ k='MADHU'
    print(len(k))=5    (len= length).
```

```
  [ l=["abc", "def", "ghi"]
    for i in ~~l~~ range(0,4,2):
        print(l[i])          (∵ i= index of list)
  → [abc
     ghi ]
```

(for-else combination only in Python).
Note: (It will come to else step only when
(entire for-loop is exhausted).

- sum of all elements in list:

```
num = [2, 3, 8, 4]
sum = 0.
for val in num:
    sum= sum+val
print (" sum is =", sum)
```
→ sum is = 17

- for-else:

```
digits = [0,5,9]
for i in digits:
    print(i)
else:
    print("No items left")
```
→
```
0
5
9
No items left
```

Rem: if for loop breaks in b/w then else won't be executed.

- while-loop:

```
sum=0
n=10
i=1
while i<=n:
    sum = sum+i
    i = i+1
print (" sum is ", sum)
```

output:
sum is 55

{ range() function always generates integer, where in numpy we have xrange() function which generates float. }

- Range() function: (generator func.)

```
range(10) → range(0, 10).
                  l.b. u.b.
list(range(10)) → [0,1,2,3,4,5,6,7,8,9]
```

(Note: upper bound is always excluded.)

→ range(0, 10, 2) — gap/jump

lower bound ↑ (0), upper ↑ (10)

~~list (range (0, len~~

```
l = ['MA', 'DH', 'UJ', 'AI', 2, 8, 9]
list(range(0, len(l), 2)
  → [0,2,4,6]
for i in range(0, len(l), 2)
        print(l[i])
  → MA
    UJ
    2
    9
```

- break and continue:

```
* for val in "MADHU":
      if val == D :
          break
      print(val)
  else:
      print("after loop")
  print("THE END")
```

Output:-

```
M
A

THE END.
```

* for val in "string":
```
for val in "string":
    if val == i:
        continue      (continue by skipping
                        current statement)
    print(val)
print("THE END")
```
output:
```
s
t
r
n
g
THE END
```

● Input:
```
name = input("what's your name?")
int(name)
type(int(input()))
 2432    (typecast)
 int
```
input be whatever it will be stored as string. later we may change by typecasting.