Data Science & AI
&
NIC - Param

Python-For Data Science

List

Lecture No.- 02

By- Pankaj Sharma Sir

Recap of Previous Lecture

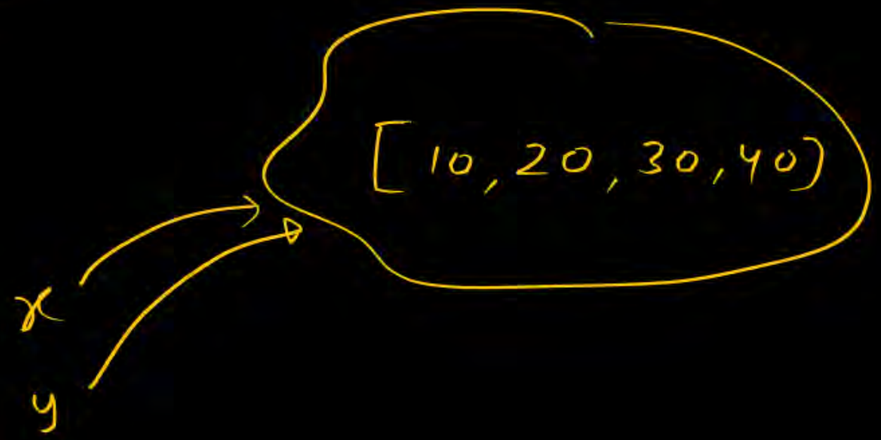
Topic List Part-01

Topics to be Covered

Topic List Part-02

Aliasing

```
x = [10, 20, 30, 40]
y = x
```



Compare

list → order ✓

l1 == l2

① elements same
② elements order
③ len

l1 = [1, 2, 3]
l2 = [3, 2, 1]
l3 = [1, 2, 3]

l1 == l2 False
l2 == l3 False
l1 == l3 True

clear()

↳ to remove all the elements

list

```
l = [1, 2, "abcd"]

l1 = [1, 2, [20, 30], "Pankaj"]
```

2d list :

```
                    1st ele        2nd ele        3rd ele
l = [ [1, 2, 3], [4, 5, 6], ["Pankaj", 7, 8] ]
```

elements in l ⇒ (3)

l[0] ⇒ 1d list
l[1]
l[2]

```
l = [ [1,2,3],
      [4,5,6],
      [7,8,9] ]
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

3 × 3

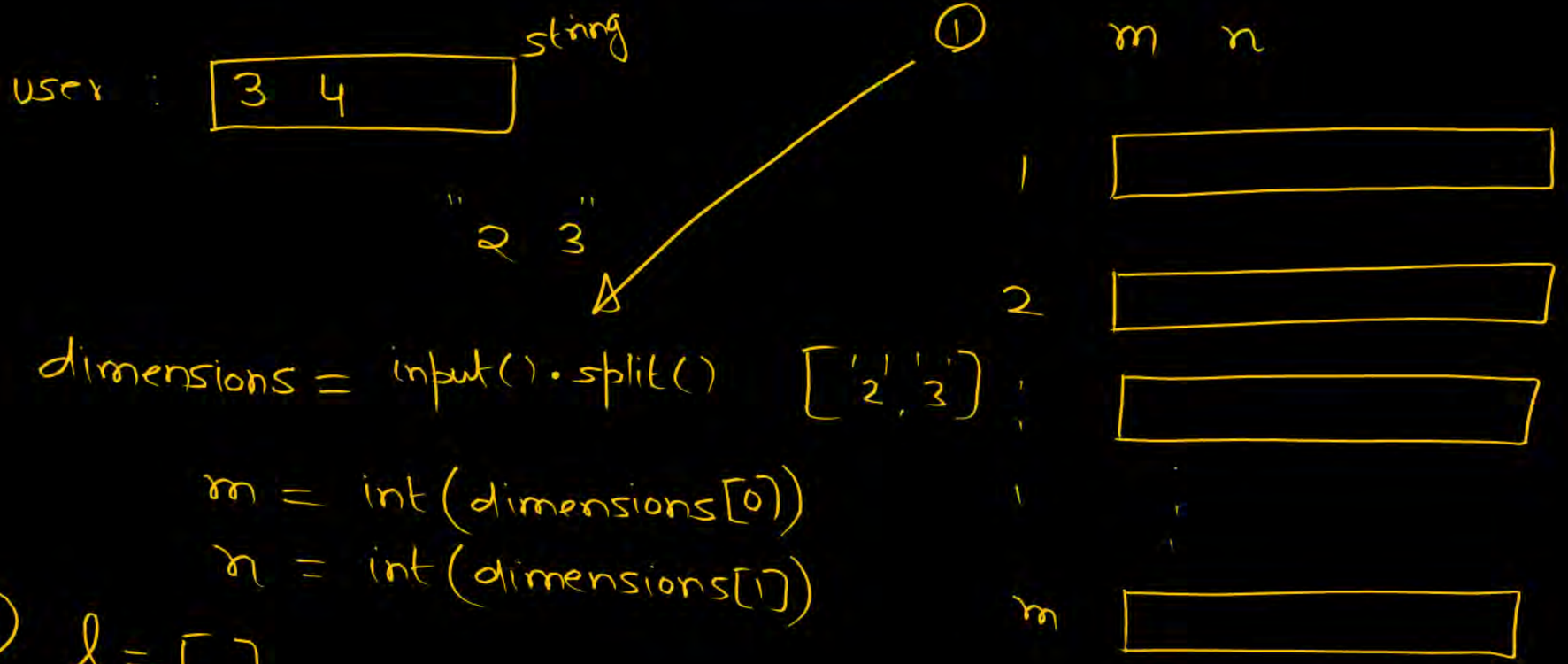
```
print( len(l) ) ⇒ 3

print( len(l[0]) )
len( [1,2,3] )   ⇒ 3
```

$m \times n$

```
l = [ [1, 2, 3, 4],
      [5, 6, 7, 8],
      [9, 10, 11] ]
```

valid in python

Problem solving



for ele in input().split()

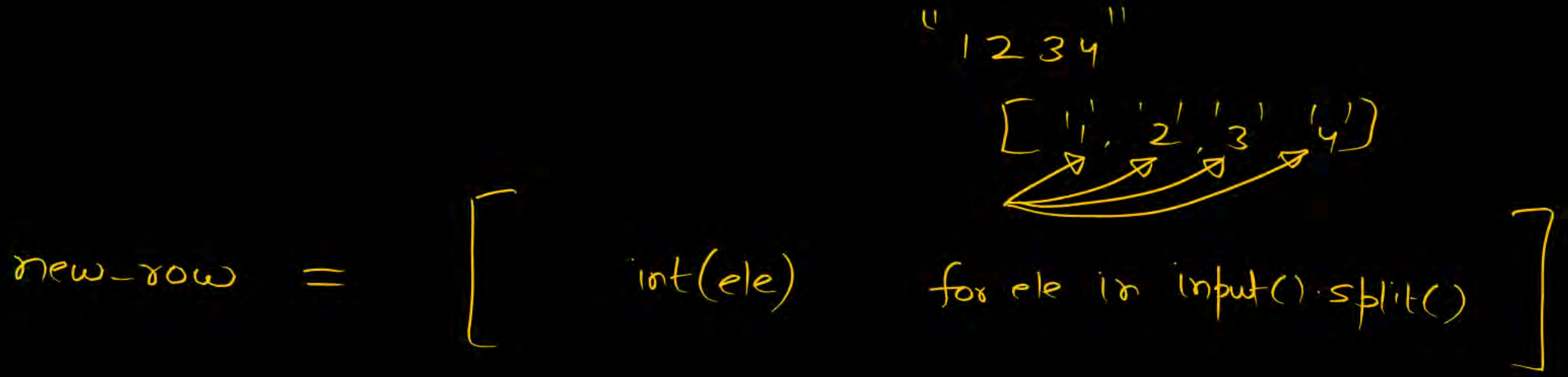
"1 2 3 4"

x = input().split()

x ⇒ ['1', '2', '3', '4']

```
l = []
for ele in x :
    l.append(int(ele))
```

l ⇒ [1, 2, 3, 4]


"1 2 3 4"
['1', '2', '3', '4']
new_row = [int(ele) for ele in input().split()]
[1,2,3,4]

(2) 1st line m n

2nd line | m×n entries |

(1) 1st line | m n |

m×n entries

1

2

⋮

2nd case

1st line: 3 4

```
1
2
3
4
5
6
7
8
9
10
11
12
```

```
dim = input().split()
m = int(dim[0])
n = int(dim[1])
```

m ⇒ 3
n ⇒ 4

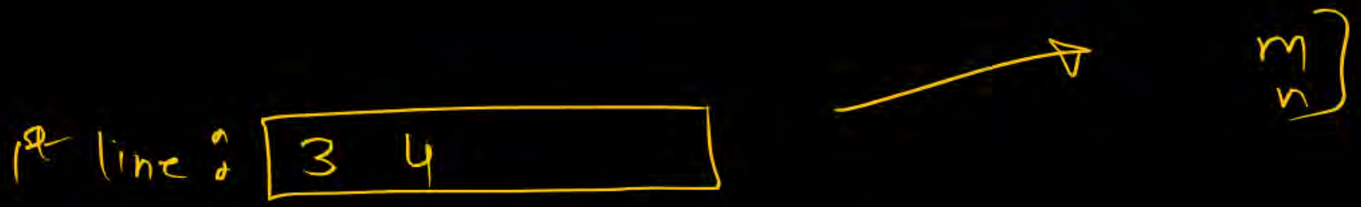
```
l = []                  # 2-d list
for i in range(m):
    new_row = []
    for j in range(n):
        ele = int(input())
        new_row.append(ele)
    l.append(new_row)
print(l)
```

4 times

Pinto Ji

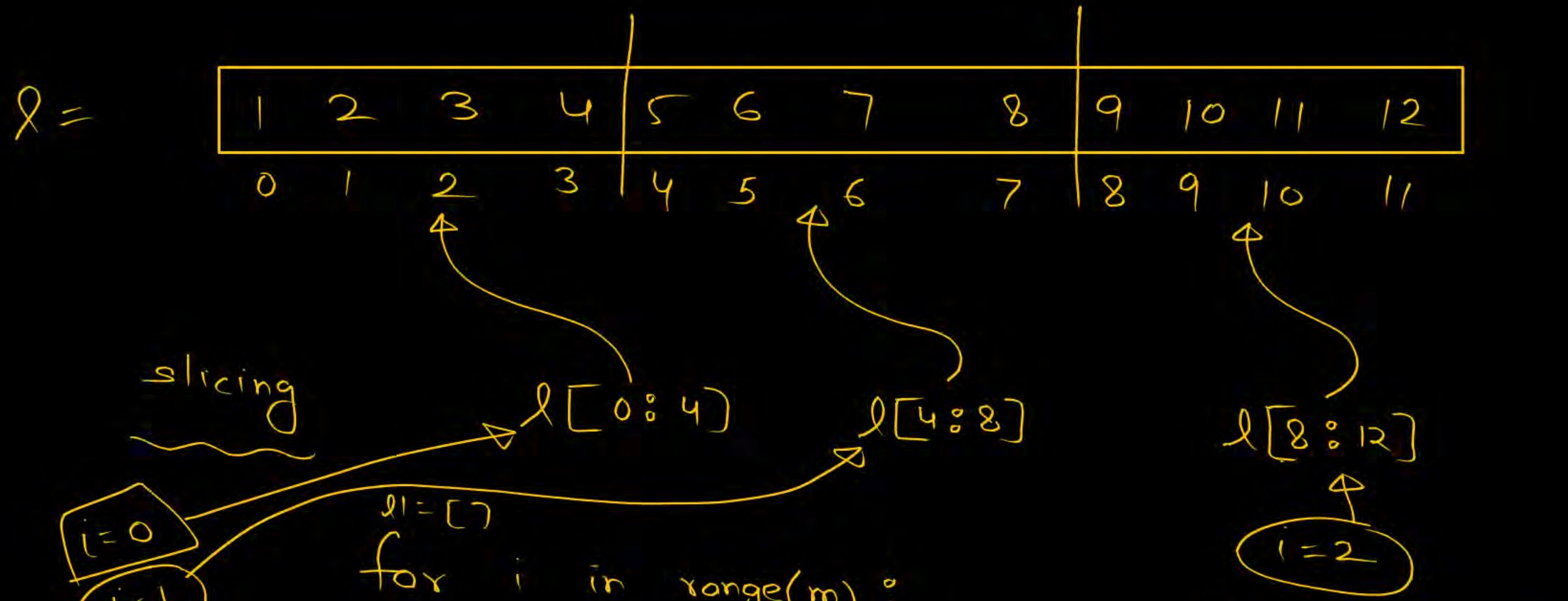
```
[ [1 2 3 4),
  [3 4 5] ]
```

[1, "Pankaj" 3 4)



l1 = [1,2,3,4,
4,5,7,8,
9,10,11,12]

```
l1 = [ int(i)  for i in input().split()]
```


l =
1 2 3 4 5 6 7 8 9 10 11 12
0 1 2 3 4 5 6 7 8 9 10 11
slicing
l[0:4]
l[4:8]
l[8:12]
i=0
i=1
i=2
l1=[]
for i in range(m):
l1.append(l[n*i : n*(i+1)])
l[begin : end]
i=0 → 0
i=1 → 4
i=2 → 8
4
8
12

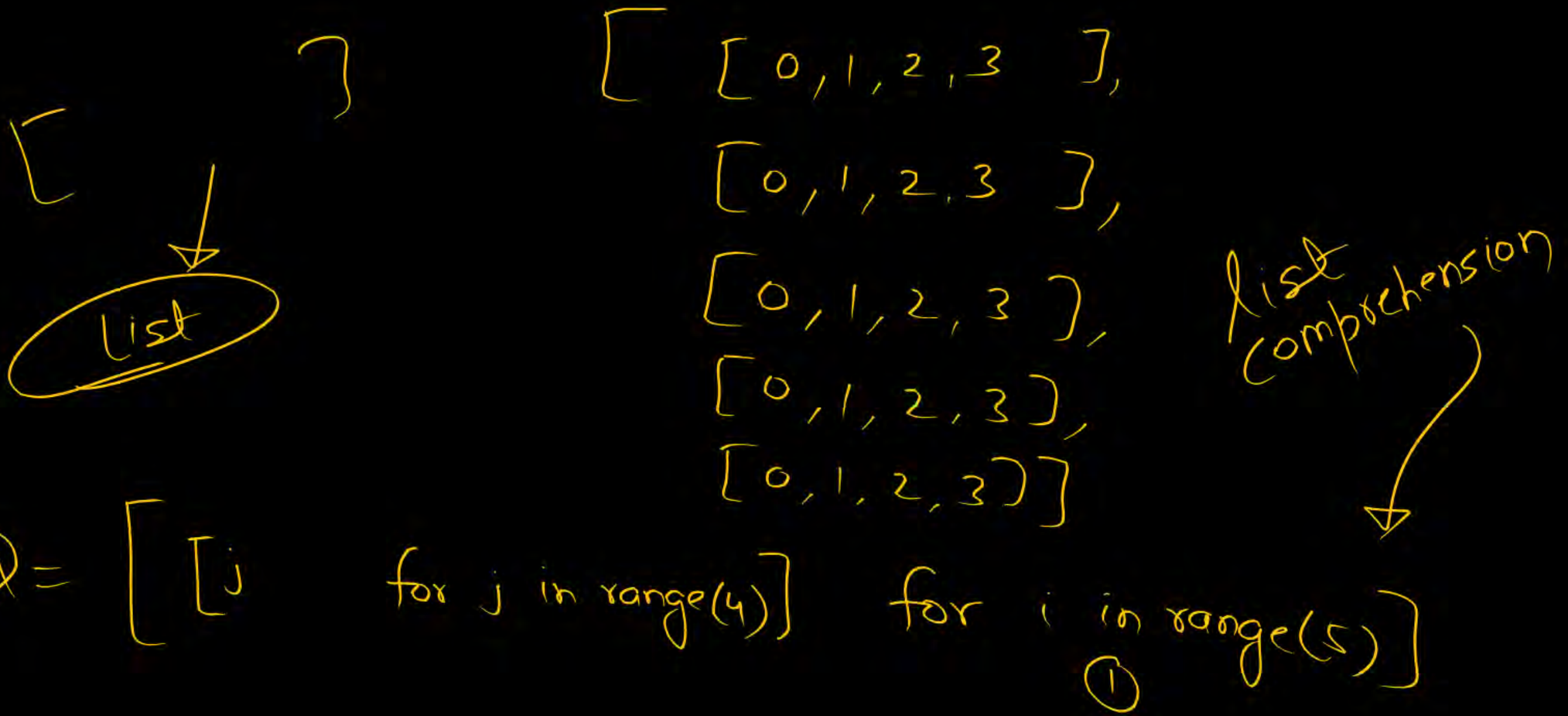
```
l = [ i  for i in range(5)]
```

```
l = [0 , 2 3 4 ]
```

2d list

```
l = [ [1, "Pankaj", 2]  for i in range(5)]
```

```
l = [ [ ]  for i in range(5)]
```


[ ]
[
list
[[0,1,2,3],
[0,1,2,3],
[0,1,2,3],
[0,1,2,3],
[0,1,2,3]]
list comprehension
l = [[j for j in range(4)] for i in range(5)]
1

Numpy
Pandas
} ⇒ 10 lecture
No promise

Theory
↘
X

Problem solving ← list, string

set
tuple
| lecture

dictionary
| lecture

functions
recursion
4 lecture

Problem solving

Review

Day 13 : python multidimensional list

In [1]:
```
l=[1,2,[10,20,"pankaj"]] #list containing another list as element
print(l)
```

```
[1, 2, [10, 20, 'pankaj']]
```

In [52]:
```
a=[[1,2,3],[4,5,6]]#2d list 2 rows and 3 column
b=[[1,2,3],[4,5]] #2d list in python
print(a)
print(b)
```

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

In [53]:
```
print(a[0])#again a list
```

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

In [54]:
```
print(a[0][1])#second element of 1st list in a
```

```
2
```

In [5]:
```
a=[[1,2,3],[4,5,6]]
print(len(a))#only 2 list/element in a
print(len(a[0]))
```

```
2
3
```

In [55]:
```
dimensions=input().split()
m=int(dimensions[0])#rows
n=int(dimensions[1])#column
l=[] #2d list
for i in range(m):#need not to worry about n
    new_row=[int(i) for i in input().split()]
    l.append(new_row)
print(l)
```

```
3 4
1 2 3 4
3 4 5 6
6 7 8 9
[[1, 2, 3, 4], [3, 4, 5, 6], [6, 7, 8, 9]]
```

In [ ]:
```
dimensions=input().split()
m=int(dimensions[0])
n=int(dimensions[1])
l=[]
for i in range(m):
    new_row=[] #for each row i am taking a empty row and fill it
    #for each row n inputs are needed
    for  j in range(n):
        ele=int(input())
        new_row.append(ele)
    l.append(new_row)
l
```

In [ ]:
```
dimensions=input().split()
m=int(dimensions[0])
n=int(dimensions[1])
l=[]
for i in range(m):
    new_row=[] #for each row i am taking a empty row and fill it
    #for each row n inputs are needed
    for  j in range(n):
        ele=int(input())
        new_row.append(ele)
    l.append(new_row)
l
```

In [ ]:
```

```

In [3]:
```
dimensions=input().split()
m=int(dimensions[0])
n=int(dimensions[1])
list_in_1d=[int(i) for i in input().split()]
#sab kuch 1d list ki form main
#m*n elements
l=[]
for i in range(m):
    begin=n*i
    end=n*(i+1)
    new_row=list_in_1d[begin:end]
    l.append(new_row)
print(l)
```

```
3 4
1 2 3 4 5 6 7 8 0 0 0 0
[[1, 2, 3, 4], [5, 6, 7, 8], [0, 0, 0, 0]]
```

In [19]:
```
#slicing same as string or any other collection
```

In [20]:
```
a=[i for i in range(5)]
a
```

Out[20]:
```
[0, 1, 2, 3, 4]
```

In [21]:
```
a=[[1,2,3,4] for i in range(5)]
print(a)
```

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

In [22]:
```
a=[[0 for j in range(5)]for i in range(5)]
print(a)
```

```
[[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]
```

In [23]:
```
a=[[j for j in range(5)]for i in range(5)]
a
```

Out[23]:
```
[[0, 1, 2, 3, 4],
 [0, 1, 2, 3, 4],
 [0, 1, 2, 3, 4],
 [0, 1, 2, 3, 4],
 [0, 1, 2, 3, 4]]
```

In [24]:
```
a=[[i for j in range(5)]for i in range(5)]
a
```

Out[24]:
```
[[0, 0, 0, 0, 0],
 [1, 1, 1, 1, 1],
 [2, 2, 2, 2, 2],
 [3, 3, 3, 3, 3],
 [4, 4, 4, 4, 4]]
```

In [ ]:
```

```

In [2]:
```
#aliasing concept
x=[10,20,30,40]
y=x
id(x)
```

Out[2]:
```
2276287718336
```

In [3]:
```
id(y)
```

Out[3]:
```
2276287718336
```

In [4]:
```
x[0]=1111
```

In [5]:
```
x
```

Out[5]:
```
[1111, 20, 30, 40]
```

In [6]:
```
y
```

Out[6]:
```
[1111, 20, 30, 40]
```

In [7]:
```
#to get rid of this problem ===>cloning
#creating a new object with same content
#1st way is slicing
#2nd way is copy()
a=[10,20,30,40,50]
b=a[:]
```

In [8]:
```
print(a)
```

```
[10, 20, 30, 40, 50]
```

In [9]:
```
print(b)
```

```
[10, 20, 30, 40, 50]
```

In [10]:
```
id(a)
```

Out[10]:
```
2276287645888
```

In [11]:
```
id(b)
```

Out[11]:
```
2276287716992
```

In [16]:
```
a[0]=1111
```

In [17]:
```
a
```

Out[17]:
```
[1111, 20, 30, 40, 50]
```

In [18]:
```
b
```

Out[18]:
```
[10, 20, 30, 40, 50]
```

In [19]:
```
x=[1,"pankaj",3,40]
y=x.copy()
```

In [20]:
```
x
```

Out[20]:
```
[1, 'pankaj', 3, 40]
```

In [21]:
```
y
```

Out[21]:
```
[1, 'pankaj', 3, 40]
```

In [22]:
```
id(x)
```

Out[22]:
```
2276287715264
```

In [23]:
```
id(y)
```

Out[23]:
```
2276287644288
```

In [24]:
```
x[0]=1111
```

In [25]:
```
x
```

Out[25]:
```
[1111, 'pankaj', 3, 40]
```

In [26]:
```
y
```

Out[26]:
```
[1, 'pankaj', 3, 40]
```

In [27]:
```
#= operator  :aliasing
#copy(),slicing : cloning
```

In [28]:
```
l1=[1,2,3,4,"pankaj"]
l1+30#error
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[28], line 2
      1 l1=[1,2,3,4,"pankaj"]
----> 2 l1+30

TypeError: can only concatenate list (not "int") to list
```

In [29]:
```
l1=[1,2,3,4,"pankaj"]
l2=[10,20,3,4,"neeraj"]
s=l1+l2 # + operator ==>same type inn general
#here list concatenation
```

In [30]:
```
s
```

Out[30]:
```
[1, 2, 3, 4, 'pankaj', 10, 20, 3, 4, 'neeraj']
```

In [34]:
```
#repeatition operator
a=[1,2,"pankaj"]
a*3
print(a*3)
print(a)
```

```
[1, 2, 'pankaj', 1, 2, 'pankaj', 1, 2, 'pankaj']
[1, 2, 'pankaj']
```

In [35]:
```
a=[1,2,"pankaj"]
a*3
a=a*3
print(a)
```

```
[1, 2, 'pankaj', 1, 2, 'pankaj', 1, 2, 'pankaj']
```

In [36]:
```
#can we compare two lists?
a=[1,2,3]
b=[3,2,1]
c=[1,2,3]
print(a==b)
```

```
False
```

In [37]:
```
print(b==c)
```

```
False
```

In [38]:
```
print(a==c)
```

```
True
```

In [39]:
```
a!=b#this is True
```

Out[39]:
```
True
```

In [40]:
```
b!=c
```

Out[40]:
```
True
```

In [41]:
```
a!=c
```

Out[41]:
```
False
```

In [42]:
```
#relational operaotrs <,<=,>,>=
a=[1,2,3]
b=[2,3,4]
a<b#1st element ke basis pe decision hoga 1<2 bcz 1st element is different==>True
```

Out[42]:
```
True
```

In [44]:
```
a=[1,2,3]
b=[1,2,4]
a<b #1,1 ==>2,2==> 3<4 True
```

Out[44]:
```
True
```

In [45]:
```
a=[1,2,3]
b=[1,2,4]
a>b
```

Out[45]:
```
False
```

In [46]:
```
l=[1,2,3]
l.clear()
```

In [ ]:
```
l
```

In [2]:
```
dimensions=input().split()
m=int(dimensions[0])
n=int(dimensions[1])
l=[]
for i in range(m):
    new_row=[] #for each row i am taking a empty row and fill it
    #for each row n inputs are needed
    for  j in range(n):
        ele=int(input())
        new_row.append(ele)
    l.append(new_row)
l
```

```
3 4
1
2
3
4
5
6
7
8
9
0
12
13
```

Out[2]:
```
[[1, 2, 3, 4], [5, 6, 7, 8], [9, 0, 12, 13]]
```

In [ ]:

In [ ]:

THANK - YOU