List Slicing

List slicing

$$\begin{array}{rccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ \text{runs} = [& 5, & 1, & 2, & 7, & 6, & 3, & 4\ ] \end{array}$$

First 5

$$\begin{array}{rccccc|cc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ \text{runs} = [& \underline{5,} & \underline{1,} & \underline{2,} & \underline{7,} & \underline{6,} & 3, & 4\ ] \end{array}$$

range(0, 5)
= [0, 1, 2, 3, 4]

List slice ⇓

$$\boxed{\text{runs}[\underset{\text{start}}{0} : \underset{\underset{\text{excluded}}{\text{end}}}{5} : \underset{\text{increment}}{1}]}$$

range(0, 5, 1)
= [0, 1, 2, 3, 4]

- → new list
- → original list remains unchanged

Cut a cake into slices ⇒ cake

Default Values

range(0, 5, 1)

range(0, 5)

range(5)

=> [0, 1, 2, 3, 4]

start = 0 default.

inc = 1 default.

end is mandatory

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

runs[0 : 5 : 1]

runs[0 : 5] => default inc = 1

=> colon helps tell it's a slice

runs[:5]

runs[5] => 3 ✗

runs[:5] => { start = 0, inc = 1 } => default

end = 5

range (5)

↑

end ✓

Does a list have a default ending point?

Yes, end of the list.

List Slicing

0 1 2 3 4 5 6 7

runs = [5, 1, 2, 7, 6, 3, 4]

len(runs) = 7

default end = len(l)

Slice ← runs [:]

default start = 0

end = len(l)

inc = 1.

len(runs) = 7

runs [0 : 7 : 1]

runs [0 : 7]

runs [: 7]

$\boxed{\text{runs} [:]}$

original list's copy

Is it the same list?

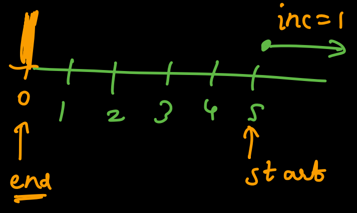
NO => only data is same

A copy/new list

=> id(runs[:]) will change.

0 1 2 3 4 5 6

runs = [5, 1, 2, 7, 6, 3, 4]

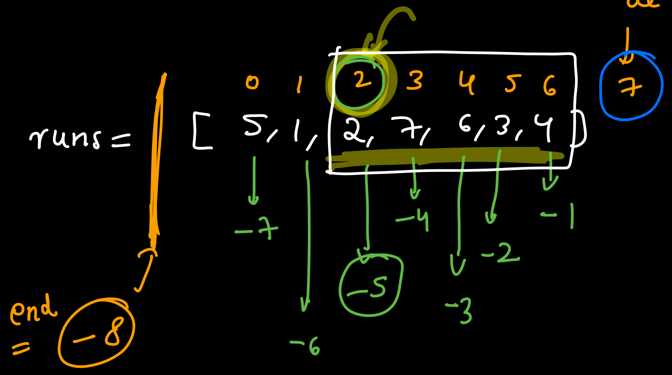


runs [5:0] → range (5, 0, 1)

=) [ ] = [ ]

Negative Indices in Slicing

default end.



7 => len(runs)

You are working on list index here

runs [-1] ← → runs [7-1]

runs [6]

NOTE:
Convert -ve index to +ve

runs[-5:]

index

start = -5 => +2 => +ve index

end = len(runs) = 7 (end of list)

inc = 1

range(-5)

= [ ] number

difference

start= 0

end= -5

inc= 1

```
         0  1  2  3  4  5  6
runs = [ 5, 1, 2, 7, 6, 3, 4 ]
        -7 -6 -5 -4 -3 -2 -1
```

convert -ve index to +ve

start = -3 ⟹ 4

end = 5

inc = 1

runs [-3 : 5 : 1]

runs[4:5:1]

[6]

~~range(-3, 5, 1) = [-3, -2, -1, 0, 1, 2, 3, 4]~~

runs[-3 : 5]   default inc = 1

[6]

---

runs[-3 :]   default inc = 1

Last 3 elements

[6, 3, 4]   default end = len(l) = 7

```
         0  1  2  3  4  5  6
runs = [ 5, 1, 2, 7, 6, 3, 4 ] ‖ 7
        -7 -6 -5 -4 -3 -2 -1
```

runs[-5:]

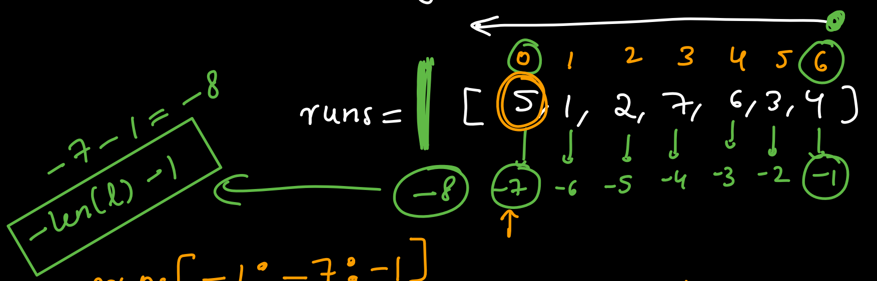
start at 5th last, go till end.

-5 ⟹ 2

```
         0  1  2  3  4  5  6
runs = [ 5, 1, 2, 7, 6, 3, 4 ] ‖ 7
        -7 -6 -5 -4 -3 -2 -1
```

[2, 7, 6, 3, 4]

Q) Given a list, print it in reverse using list slicing.



runs[-1:-7:-1]

↳ [4, 3, 6, 7, 2, 1, ~~5~~]

start = -1 ⇒ 6

end = -7 ⇒ 0

inc = -1

+ve inc →

-ve inc ←

runs[-1:-8:-1]

[4, 3, 6, 7, 2, 1, 5]

-len(runs)-1

Default end when inc is negative. (going right to left)

NOTE: We can check using range with start & end +ve

range(6, 0, -1)

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

runs[-1:-8:-1]

X runs[-1:-1] X

start = -1

end = -1

inc = +1

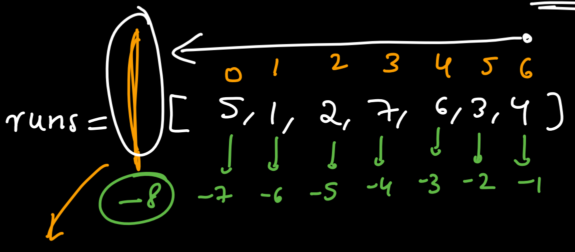
runs[-1::-1]

needed

start=-1

end=

inc=-1



default end.

Can we skip the start? when inc = -ve

runs [: : -1]

[4, 3, 6, 7, 2, 1, 5]

default start = -1

Last ele.

default end = end in the reverse

-len(l) - 1

inc = -1

same

-len(l)-1

runs [-1 : -8 : -1]

runs [-1 : : -1]

runs [: : -1]

When inc > 0 :

index

default start = 0

default end = len(l)

default inc = 1

runs[:]

Copy

When inc < 0 :

index

default start = -1

default end = -len(l)-1

inc < 0

must be there

runs[: : -1]

reverse