	0	1	2	3	4	5	6	7	8	9	10	11
L	10	20	30	40	50	60	70	80	90	100	110	120

L[i:j] $\quad i < j \quad i \geq j$

① $i < j \rightarrow len(L[i:j]) > 0$

② $i \geq j \rightarrow len(L[i:j]) = 0$

③ $k > 0$ and $i < j \rightarrow len(L[i:j:k]) > 0$

④ $k > 0$ and $i \geq j \rightarrow len(L[i:j:k]) = 0$.

New piece of information:

① Step can be negative as well.

② Sign of the step count determines:

(A) the direction of traversal

(B) Stopping point.

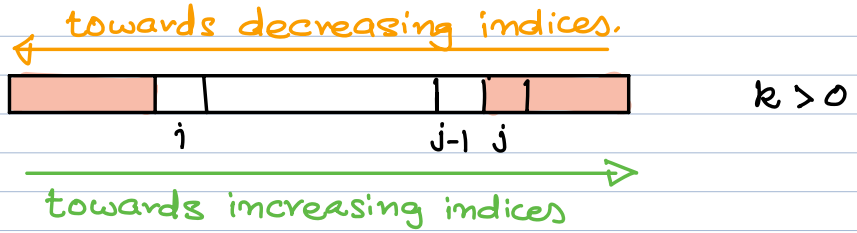
Case-I: Step count k is positive.

(A) Direction of traversal:

towards increasing indices.

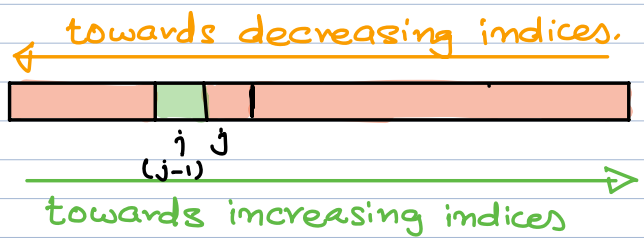
(B) Stopping point:

j-1.

① 

$k > 0$

(2)

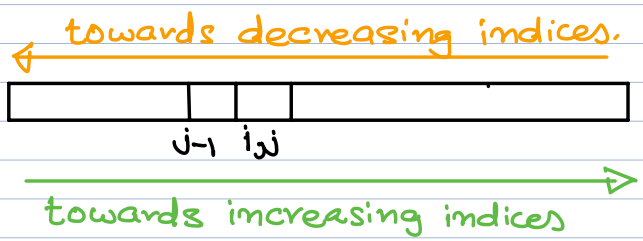


$k > 0$

L [2 : 3 : 1] ⟶ only one element will be selected.

2
3
4

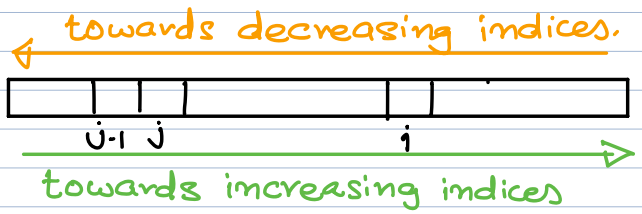
(3)



$(i == j)$

$k > 0$

(4)



Case-II : Step count $k < 0$.

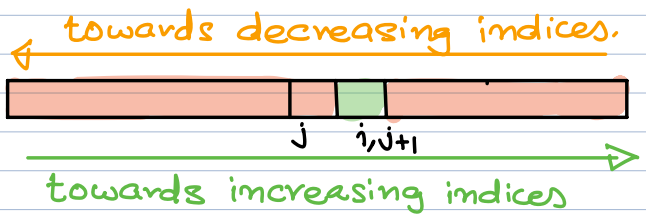
(A) Direction of traversal ⟶ towards decreasing indices.

(B) Stopping Point : j+1

(1)

$k < 0$

(2)

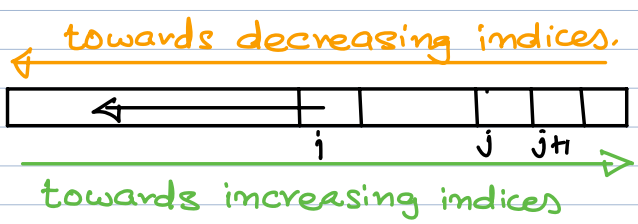


$k < 0$

(3)

$k < 0$

(4)



$k < 0$

Updated Equations:

$k > 0$ and $i < j \longrightarrow \text{len}(L[i:j:k]) > 0$

$k > 0$ and $i \geq j \longrightarrow \text{len}(L[i:j:k]) = 0$

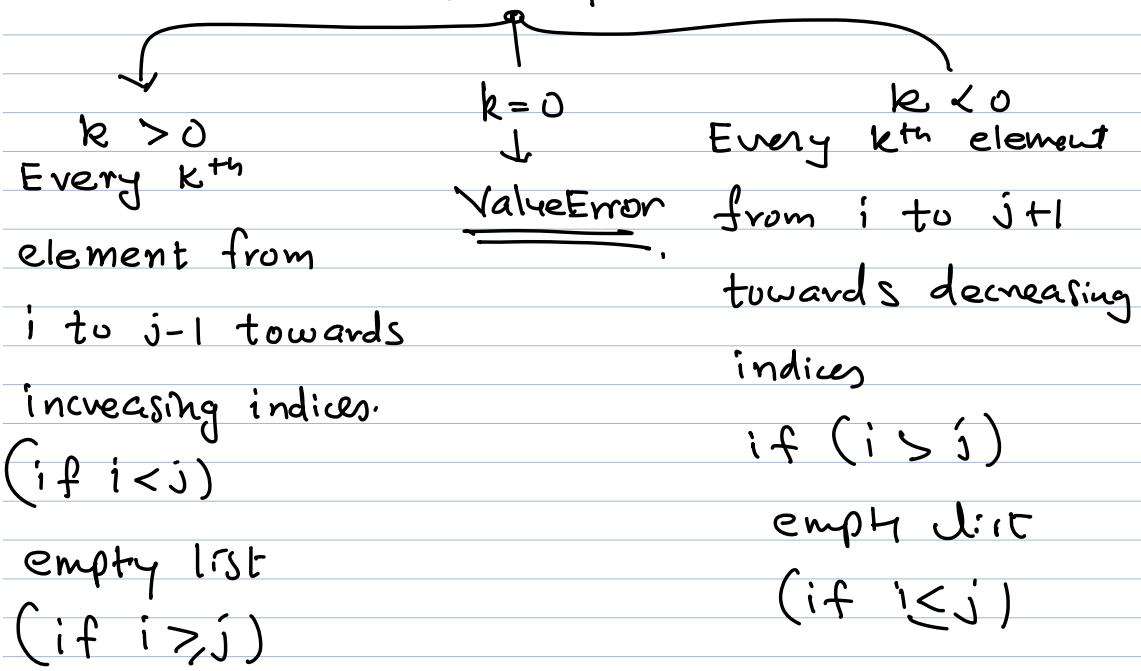
$k < 0$ and $i > j \longrightarrow \text{len}(L[i:j:k]) > 0$

$k < 0$ and $i \leq j \longrightarrow \text{len}(L[i:j:k]) = 0.$

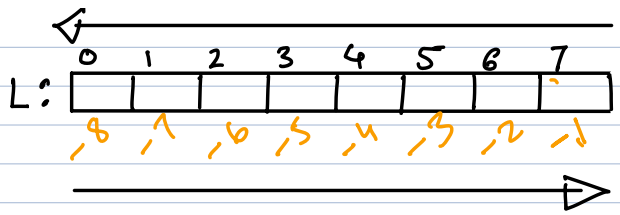
$\text{len}(L[i:j:k]) > 0 \longrightarrow (k > 0 \text{ and } i < j) \text{ or } (k < 0 \text{ and } i > j)$

$\text{len}(L[i:j:k]) = 0 \longrightarrow (k > 0 \text{ and } i \geq j) \text{ or } (k < 0 \text{ and } i \leq j)$

General Algorithm for Slice Computation.




```
def get_slice(L: list, i:int, j:int, k:int):
    if k == 0:
        raise ValueError('Step count cannot be zero')
    result = []
    if k > 0:
        if i >= j:
            return result # return empty list
        else: # Every kth element from i to j - 1 towards increasing indices
            r = i
            while r < j:
                result.append(L[r])
                r = r + k
    else:
        if i <= j:
            return result # return empty list
        else: # Every kth element from i to j + 1 towards decreasing indices
            r = i
            while r > j:
                result.append(L[r])
                r = r - k
    return result
```



let L be any list

Positive valid index range : 0 to len(L)-1

Negative valid index range : -len(L) to -1

L[i] $\quad -len(L) \leq i < len(L)$