List operations

For the following examples, we assume that `l1` and `l2` are lists, `x`, `i`, `j`, `k`, `n` are integers.

`l1 = [10, 20, 30, 40, 50]` and `l2 = [60, 70, 80, 60]`

`x in l1`	Check if the list `l1` contains item `x`.
`x not in l2`	Check if list `l1` does not contain item `x`.
`l1 + l2`	Concatenate the lists `l1` and `l2`. Creates a new list containing the items from `l1` and `l2`.
`l1 * 5`	Repeat the list `l1` 5 times.
`l1[i]`	Get the item at index `i`. Example `l1[2]` is 30.
`l1[i:j]`	List slicing. Get the items from index `i` up to index `j` (excluding `j`) as a List. An example `l1[0:2]` is `[10, 20]`

`l1[i:j:k]`	List slicing with step. Returns a List with the items from index `i` up to index j taking every k-th item. An example `l1[0:4:2]` is [10, 30].
`len(l1)`	Returns a count of total items in a list.
`l2.count(60)`	Returns the number of times a particular item (60) appears in a list. The answer is 2.
`l1.index(30)`	Returns the index number of a particular item (30) in a list. The answer is 2.
`l1.index(30, 2, 5)`	Returns the index number of a particular item (30) in a list. But search Returns the item with maximum value from a list. The answer is 60 only from index number 2 to 5.
`min(l1)`	Returns the item with a minimum value from a list. The answer is 10.
`max(l1)`	Returns the item with maximum value from a list. The answer is 60.

`l1.append(100)`	Add item at the end of the list
`l1.append([2, 5, 7])`	Append the nested list at the end
`l1[2] = 40`	Modify the item present at index 2
`l1.remove(40)`	Removes the first occurrence of item 40 from the list.
`pop(2)`	Removes and returns the item at index 2 from the list.
`l1.clear()`	Make list empty
`l3= l1.copy()`	Copy l1 into l2

Tuple operations

For the following examples, we assume that `t1` and `t2` are tuples, `x`, `i`, `j`, `k`, `n` are integers.

`t1 = (10, 20, 30, 40, 50)` and `t2 = (60, 70, 80, 60)`

Operation	Description
`x in t1`	Check if the tuple `t1` contains the item `x`.

Operation	Description
`x not in t2`	Check if the tuple `t1` does not contain the item `x`.
`t1 + t2`	Concatenate the tuples `t1` and `t2`. Creates a new tuple containing the items from `t1` and `t2`.
`t1 * 5`	Repeat the tuple `t1` 5 times.
`t1[i]`	Get the item at the index `i`. Example, `t1[2]` is 30
`t1[i:j]`	Tuple slicing. Get the items from index `i` up to index `j` (excluding j) as a tuple. An example `t1[0:2]` is (10, 20)
`t1[i:j:k]`	Tuple slicing with step. Return a tuple with the items from index `i` up to index j taking every k-th item. An example `t1[0:4:2]` is (10, 30)
`len(t1)`	Returns a count of total items in a tuple
`t2.count(60)`	Returns the number of times a particular item (60) appears in a tuple. Answer is 2
`t1.index(30)`	Returns the index number of a particular item(30) in a tuple. Answer is 2

Operation	Description
`t1.index(40, 2, 5)`	Returns the index number of a particular item(30) in a tuple. But search only from index number 2 to 5.
`min(t1)`	Returns the item with a minimum value from a tuple
`max`(t1)	Returns the item with maximum value from a tuple