```
from collections import Counter, defaultdict, ....
collections.Counter(list).most_common()
collections.defaultdict(int) or (list) or (set) or (?)

"""to use heapq, no need to define new class with __lt__ function
just use tuple (major, minor, ...). tuple has __lt__ function already"""
from heapq import heappush, heappop, heapify
heap = [....]
heapify(heap)
"""WRONG!!"""  heap = heapify([....])

len(dict)

abs(-1)
math.inf, -math.inf

queue: deque.append, deque.popleft
stack:  queue.append, queue.pop

global_var = [0,]

a != b

clone/copy a list: lst.copy()

alter i-th char in string:  s[:i] + '?' + s[i+1:]

dict comprehension: {a:b for a, b in zip(A, B)}

all(), any()

put generator in parentheses: (generator)


str.isnumeric()
''.isnumeric() == False

intervals.sort(key=lambda x: x.start)

from fractions import gcd
from itertools import product, islice, combinations,
combinations_with_replacement, permutations
```

```
itertools.product(*[list, list])     # returns tuple
from functools import reduce
```