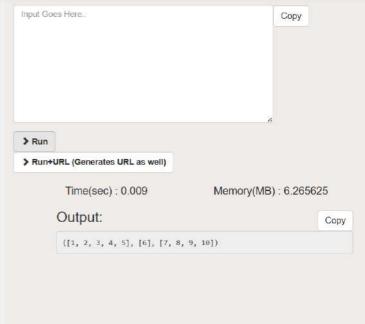
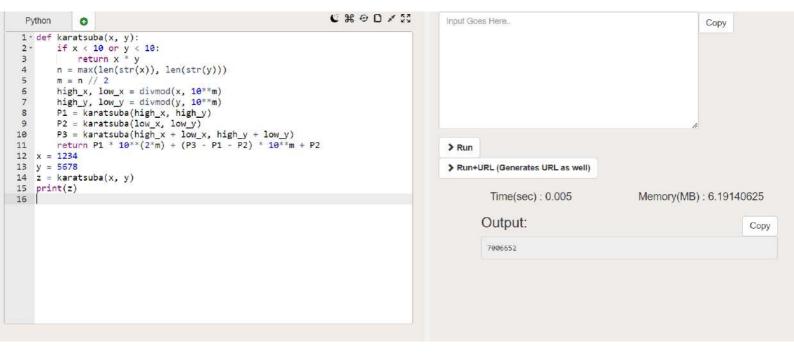
```
C # 0 D / X
  Python o
                                                                                                                             Input Goes Here..
                                                                                                                                                                                                         Сору
 1 arr = [12, 3, 5, 7, 19]
2 k = 2
 3 def select(arr, k):
4 if len(arr) <= 5:</pre>
 4 -
5
6
7
            return sorted(arr)[k]
medians = [sorted(arr[i:i + 5])[len(arr[i:i + 5]) // 2] for i in rang
pivot = select(medians, len(medians) // 2)
lows = [x for x in arr if x < pivot]
highs = [x for x in arr if x > pivot]
pivots = [x for x in arr if x == pivot]
if k < len(lows):

return select(lows k)
  8
  9
10
11 -
                                                                                                                              > Run
> Run+URL (Generates URL as well)
                                                                                                                                          Time(sec): 0.012
                                                                                                                                                                                      Memory(MB): 6.171875
                                                                                                                                        Output:
                                                                                                                                                                                                                     Сору
```

```
C# 0 D x 23
   Python
                           0
 1 def partition(arr, pivot):
2 low = [x for x in arr if x < pivot]
3 high = [x for x in arr if x > pivot]
4 pivots = [x for x in arr if x == pivot]
5 return low, pivots, high
6 def find_median(arr):
7 arr cort()
  7 arr.sort()
8 return arr[len(arr) // 2]
9 def median_of_medians(arr, k):
10 if len(arr) <= 5:
11 arr.sort()
10 -
11
                                                                                                                                                                                           > Run
                 return arr[k-1]
sublists = [arr[i:i+5] for i in range(0, len(arr), 5)]
medians = [find_median(sublist) for sublist in sublists]
pivot = median_of_medians(medians, len(medians) // 2 + 1)
low, pivots, high = partition(arr, pivot)
12
13
14
15
16
                  if k <= len(low):</pre>
17 -
                  return median_of_medians(low, k)
elif k <= len(low) + len(pivots):
    return pivots[0]
18
                                                                                                                                                                                                          Output:
19 -
20
21 -
                  else:
22 return median_of_medians(high, k - len(low) - len(pivots))
23 arr1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
24 k1 = 6
25 print(partition(arr1,k1))
```





Сору

```
C # 0 D / 23
  Python
                   0
 1 from itertools import combinations
 2 def subset_max_sum(1, r, s):
3 largest_subset = None
 4 -
            for m in range(1, len(1) + 1):
            for m in range(1, len(1) + 1):
    for sets in combinations(1, m):
        if sum(sets) == s:
            if largest_subset is None or len(sets) > len(largest_subset)
            largest_subset = sets
        for p in range(1, len(r) + 1):
        for subset in combinations(r, p):
        if sum(subset) == s:
 6 - 7 -
  8
10 -
> Run
12 -
13
16 s = 5
17 n = len(a)
18 l = a[:n//2]
19 r = a[n//2:]
20 print(subset_max_sum(1, r, s))
```

