In [9]:
```
def pclosest(points,k):
    points.sort(key=lambda k: k[0]**2+ k[1]**2)
    return points[:k]
points= [[1,3],[-2,2],[5,8],[0,1]]
k=2
print(pclosest(points,k))
```

```
[[0, 1], [-2, 2]]
```

In [10]:
```
def partition(arr, pivot):
    low = [x for x in arr if x < pivot]
    high = [x for x in arr if x > pivot]
    pivot_count = len(arr) - len(low) - len(high)
    return low, pivot_count, high
def select(arr, k):
    if len(arr) <= 5:
        return sorted(arr)[k]
    medians = [sorted(arr[i:i+5])[len(arr[i:i+5]) // 2] for i in range(0, le
    pivot = select(medians, len(medians) // 2)

    low, pivot_count, high = partition(arr, pivot)

    if k < len(low):
        return select(low, k)
    elif k < len(low) + pivot_count:
        return pivot
    else:
        return select(high, k - len(low) - pivot_count)
def median_of_medians(arr, k):
    return select(arr, k - 1)
arr = [12, 3, 5, 7, 4, 19, 26]
k = 3
print(median_of_medians(arr, k))
```

```
5
```

In [11]:
```
def four_sum_count(A, B, C, D):
    sum_AB = {}
    for a in A:
        for b in B:
            if a + b in sum_AB:
                sum_AB[a + b] += 1
            else:
                sum_AB[a + b] = 1
    count = 0
    for c in C:
        for d in D:
            target = -(c + d)
            if target in sum_AB:
                count += sum_AB[target]
    return count
A = [1, 2]
B = [-2, -1]
C = [-1, 2]
D = [0, 2]
print(four_sum_count(A, B, C, D))
```

```
2
```

In [ ]: