Problem 2

Algorithm:

1.First we are going to use Divide and conquer to divide the list L into two groups: Left and Right. Keep doing it until we reach one single element then we will return that single element.

2. Take out the largest subset in Left and test the walkie-talkie with all the subsets in Right, if they are mutually compatible, merge them into the same subset.

3. Take out the largest subset in Right and test it with all the subsets in Left, if there are two subsets mutually compatible, merge them into the same subset

4. When we do the test between subsets, we only compare the first element in the subset.

5. Put the two subsets we get from step 2 and step 3 into a list, and put all rest subset from left and right into this list as well, return it.

6. After we finish the divide and conquer, the biggest subset in the returned list is the solution

```
recursive(list):
    if the size(list) == 1:
        return list
    else:
        n = size(list)/2
        Right = recursive(list[n:])
        Left = recursive(list[0:n])
        L = the largest subset in Left
        R = the largest subset in Right
        for element in right:
            if L[0] is mutually compatible with element[0]:
                add element to L
                delete element in right

        for element in Left:
            if R[0] is mutually compatible with element[0]:
                add element to R
                delete element in Left
```

result = Left + Right
return result

Proof:

I use Divide and conquer algorithm, and each time i just merge subsets that are mutually compatible, and to the end, i will label all the walkie-talkies with different subset. We only need the biggest subset among them, so just return the subset with the biggest size at the end.

Analyze:

It is Divide and conquer algorithm, and for each combining time, we only use a linear-time operation, therefore, the total time will be in Theta(nlogn)

Problem 3

Algorithm:

1. we need to define a convolution matrix ab, where a = [n,n-1,n-2,.....3,2,1], and b = [1, $\frac{1}{4}$, $\frac{1}{9}$, $\frac{1}{16}$,....., $\frac{1}{n-2}$, $\frac{1}{n-1}$]
2. formula looks like this: Fj = C*Qj*($\frac{Qa}{(j-a)^2}$ + $\frac{Qb}{(j-b)^2}$ +....-$\frac{Qy}{(j-y)^2}$ -$\frac{Qz}{(j-z)^2}$)
3. Fj = C*q_j * (the Number on the Diagonal Starting at j-1 that points down\
   minus Numbers on the Diagonal starting at j+1 that points up)
4. Use a for-loop that counts i from 1 to n, and every time in the for-loop, we just store Fi in the list. Finally, just return the list which contains all the forces Fi.

For example:

we are going to operate on 7 particles, which means n=7, then we need to construct a convolution matrix:

a = [$q_7, q_6, q_5, q_4, q_3, q_2, q_1$] b = [1, $\frac{1}{4}$, $\frac{1}{9}$, $\frac{1}{16}$, $\frac{1}{25}$, $\frac{1}{36}$]

	1	$\frac{1}{4}$	$\frac{1}{9}$	$\frac{1}{16}$	$\frac{1}{25}$	$\frac{1}{36}$
q_7	$\frac{7}{1}$	$\frac{7}{4}$	$\frac{7}{9}$	$\frac{7}{16}$	$\frac{7}{25}$	$\frac{7}{36}$
q_6	$\frac{6}{1}$	$\frac{6}{4}$	$\frac{6}{9}$	$\frac{6}{16}$	$\frac{6}{25}$	$\frac{6}{36}$
q_5	$\frac{5}{1}$	$\frac{5}{4}$	$\frac{5}{9}$	$\frac{5}{16}$	$\frac{5}{25}$	$\frac{5}{36}$
q_4	$\frac{4}{1}$	$\frac{4}{4}$	$\frac{4}{9}$	$\frac{4}{16}$	$\frac{4}{25}$	$\frac{4}{36}$
q_3	$\frac{3}{1}$	$\frac{3}{4}$	$\frac{3}{9}$	$\frac{3}{16}$	$\frac{3}{25}$	$\frac{3}{36}$
q_2	$\frac{2}{1}$	$\frac{2}{4}$	$\frac{2}{9}$	$\frac{2}{16}$	$\frac{2}{25}$	$\frac{2}{36}$
q_1	$\frac{1}{1}$	$\frac{1}{4}$	$\frac{1}{9}$	$\frac{1}{16}$	$\frac{1}{25}$	$\frac{1}{36}$

Now , If we want to calculate F4, we will get following:

F4 = C* q_4 * (Numbers on the Diagonal starting at 5 that points up \
minus the Number on the Diagonal Starting at 3 that points down)

Number on the Diagonal Starting at 3 that points down are the number s in Blue
Numbers on the Diagonal starting at 5 that points up are the numbers in RED

Therefore, F4 = C* q_4 *($\frac{q1}{9}$ + $\frac{q2}{4}$ + $\frac{q3}{1}$ - $\frac{q5}{1}$ - $\frac{q6}{4}$ - $\frac{q7}{9}$)

Proof:

we can think this as a product by two polynomials, and a convolution matrix will give us all the combination products for given length. Then we just need to pick out the one we need.

Analyze:

We assume that constructing a convolution matrix in Theta (nlogn), and take out the element from convolution matrix doesn't cost much time. Then we just need a for-loop that counts from 1 to n, and collect all the Force and return it. So the total running time is still in Theta (nlogn)

Name : Kebi Hong