**Assignment 2**

z5142340 Haiyu LYU

**Question1:**

Let , we have that:

Because the degree of is 2 and the degree of is 3. So, we can multiply two polynomials using only 6 multiplications.

Let be the coefficient of the polynomial, then we have that:

**Question2:**

**(a)**.

Because , and make up the real part (2 multiplications). So imaginary part needs 1 multiplication. Thus, we can use only 3 real number multiplications.

**(b)**.

Because is the real part of and it just uses 1 multiplication. For imaginary part, since , we can only use 1 multiplication to find and then we can get by addition. Thus, we compute the product using 2 multiplications.

**(c)**.

We can evaluate using 3 multiplications using part **(a)**. Then square the result using 2 multiplications using part **(b)**. Therefore, 5 real number multiplications are needed.

**Question3:**

**(a).**

Assume that we have 2 polynomials called and . The product of and has degree at . In order to determine it, we need values at point. We can use FFT to evaluate and  at points. Then we multiply the results of evaluation pointwise. So we got equations. Then we need to use inverse FFT to retrieve in coefficient form. Therefore, run this algorithm in .

**(b).**

**(i).**

Let

It looks like simple recursion (base case is ).

Because , the degree of and is always less than S. For each multiplication, the time complexity may be using FFT. We have k multiplications, so the total time is .

**(ii).**

Using divide-and-conquer can reduce the time complexity.

At the first, we can make 2 polynomials as a pair, then we need calculate the product of the pair in . For example, we can get the product ,.

For each , we can still set two adjacent C as a pair. Then we can also get the product of them.

Repeat the above algorithm until we finally get one result. This looks like a tree and the height of tree is .

Therefore, we think is needed for this algorithm.

**Question4:**

**(a)**.

To proof , we use mathematical induction:

Because and .

So

Thus, by mathematical induction, for all integer n>1,

**(b)**.

Let is matrix of the Fibonacci numbers.

……

We can clearly see that seems like a tree and the tree height is . Because we calculate the multiplication of two matrix in any level in So the time complexity of is .

**Question5:**

**(a).**

To find those choice of leaders, we only consider the case that the height of giant is less than or equal to T. Firstly we loop through the array H and find the leader that satisfies our condition (height <=T). Record this leader and jump to the next K. If the conditions are still met, then continue to record this leader. Repeat this calculation until you have traversed the entire array. We compare whether the number of leaders recorded is equal to L. If it equal to L, the leader who satisfies the condition exists. The complexity of this algorithm is .

**(b).**

We must find the ideal T, which is the maximum value of T. We can think of T as monotonic, because if there is a larger T that satisfies the condition, then a smaller T must also satisfy the condition. So we sort the possible T as a list. Then our strategy is to use the binary search to determine the maximum value of T. We can find a suitable T by binary search, and then use to verify that T meets the condition. The total complexity is