Lab 10

1. Below, the BinarySearch and Recursive Fibonacci algorithms are shown. In each case, what are the subproblems? Why do we say that the subproblems of BinarySearch *do not overlap* and the subproblems of Recursive Fibonacci *overlap?* Explain.

```
Algorithm binSearch(A, x, lower, upper)
    Input: Already sorted array A of size n, value x to be
           searched for in array section A[lower]..A[upper]
    Output: true or false

 if lower > upper then return false
 mid ← (upper + lower)/2
 if x = A[mid] then return true
 if x < A[mid] then
       return binSearch(A, x, lower, mid – 1)
 else
       return binSearch(A, x, mid + 1, upper)
```

```
Algorithm fib(n)
    Input: a natural number n
    Output: F(n)

 if (n = 0 || n = 1) then return n

 return fib(n-1) + fib(n-2)
```

2. Consider the following instance of the Edit Distance problem: EditDistance(“maple”, “kale”). Taking the iterative dynamic programming approach to solve this problem, fill out the values in the table.

D	“”	“k”	“ka”	“kal”	“kale”
“”					
“m”					
“ma”					
“map”					
“mapl”					
“maple”					

3. *(Interview Question)* Devise a dynamic programming solution for the following problem:

   Given two strings, find the length of longest subsequence that they share in common.

   Different between substring and subsequence:
   Substring: the characters in a substring of S must occur contiguously in S.
   Subsequence: the characters can be interspersed with gaps.
   For example: Given two Strings - “regular” and “ruler”, your algorithm should output 4.

4. *(Optional Interview Question)* Devise a dynamic programming solution for the following problem:

   Given a positive integer n, find the least number of perfect square numbers which sum to n.
   (Perfect square numbers are 1, 4, 9, 16, 25, 36, 49, …)
   For example, given n = 12, return 3; (12 = 4 + 4 + 4)
   Given n = 13, return 2; (13 = 4 + 9)
   Given n = 67 return 3; (67 = 49 + 9 + 9)