1.(a) At first, we have an input string S, a front index is equal to 0, and a back index is equal to 1.

```
recurValid ( S , front , back ) :
    if back is equal to the length of S :
        return dict ( S [front : back] )
    else if dict ( S [front : back] ) is True :
        return recurValid ( S , front , back + 1 ) or
               recurValid ( S , back , back + 1 )
    else dict ( S [front : back] ) is False :
        return recurValid ( S , front , back + 1 )
```

(b)
```
dpValid (S) :
    n is the length of string S
    construct a dp table with n rows and n+1 columns with
    None filled initially
    for every element j in range (1, n+1) :
        dp[0][j] = dict ( string[0 : j])
    for every element i in range (1, n) :
        for every element j in range ( i+1 , n+1) :
            if dict (string [i : j]) is true :
                dp[i][j] = dp[i-1][j] or dp[i-1][i]
            else dict (string [i : j]) is false :
                dp[i][j] = dp[i-1][j]

    return dp[n-1][n]
```

(c) We first have an empty solution array
Then we look into our dp table and set index $j = n$ initially (the dp table has n rows and $n+1$ columns)

while $j > 0$:
    look at the jth column
    for every element i in range $(0, n)$:
        if dp[i][j] is True:
            append i into our solution array, stop search and jump to the ith column ($j = i$) and start search again

    if all grid points in this column is False, stop the whole process and indicate that input string S is an invalid text document

If we successfully arrive at the first column (0th), we can say that input string S is a valid text document
Reverse our solution array and assume its element:
$[a1, a2, \cdots, an]$ and $a1 = 0$ is inevitable

Now we can reconstruct our String S by dividing it into several valid words (remember n is the length of S):
S[a1:a2], S[a2:a3], $\cdots$, S[an-1 : an], S[an : n]

(d) Since front < back is required in our algorithm, we did not fill a triangle area in the dp table where front ≥ back
So the running time: $n \times (n+1) - \frac{1}{2} \times n \times n$
$= \frac{1}{2}n^2 + n$
$\in O(n^2)$ (n is the length of string S)