a) [50 points] Give an efficient algorithm that takes strings s, x, and yand decides if sis an interweaving of xand y. Derive the computational complexity of your algorithm

Pseudocode and time complexity table

```
def boolean isInterleaved(String A, String B, String S){
    int m = A.length(); int n = B.length();
    boolean il[][] = new boolean[][];
//two-dimensional array is used to store the DP process result. il[i][j] means that
//whether or not S is a interweave of ith chars of A and j-th chars of B
    if ((m + n) != S.length())
        return false;
//m+n != S.length denifnately not result
    for(int i = 0; i <= m; i++){
        for(int j = 0; j <= n; j++){
            if (i == 0 && j == 0) {
                il [i][j] = true;
            }
            else if (i == 0){
                if (B[j – 1] ==S.[j – 1]) {
                    il [i][j] = il [i][j - 1];
                }
            }
            else if (j == 0)
            {
                if (A[i – 1]==S [i – 1]) {
                    il [i][j] = il [i - 1][j];
                }
            }
            else if (A [i – 1] == S. [i + j – 1] and B [j – 1] != [i + j – 1]) {
                    il [i][j] = il [i - 1][j];
            } else if (A[i – 1] !=S [i + j – 1] and B [j – 1] ==S [i + j – 1]) {
                    il [i][j] = il [i][j - 1];
            } else if (A[i – 1] ==S[i + j – 1] and [j – 1]==S[i + j – 1]) {
                    il [i][j] = (il [i - 1][j] || il [i][j - 1]);
            }
        }
    }
    return il [m][n];
}
```

Cost time,

c m

c n·m

c 1

c 1

c (n-1) m

c 1

c n·(m-1)

c (n-1) m

c (m-1) n

c (n-1)(m-1)

C n m

So the time complexity should be O(n*m)

Assume the length of s is |s|

```
def main(str S, X, Y)
//X and Y should be x' and y' in the problem description
    for (int i=1;i<S.length();i++){
        int j = S.length()-i;
        if (isInterleaved(X[0:i], Y[0:i],S)==true){
            return true
            break;
```

cost time

c s

c s

c s · O(nm)

c s

In the for loop, isInterleaved is called |s| times, and each isInterleaved time complexity is O(n*m), so the time complexity should be O(|s|*n*m)

Because in my code, in the main function, in each call of isInterleaved, n+m=|s|, so in total, the time complexity is $O((|s|)^2)+ O((|s|)^3)= O((|s|)^3)$

Correctness:

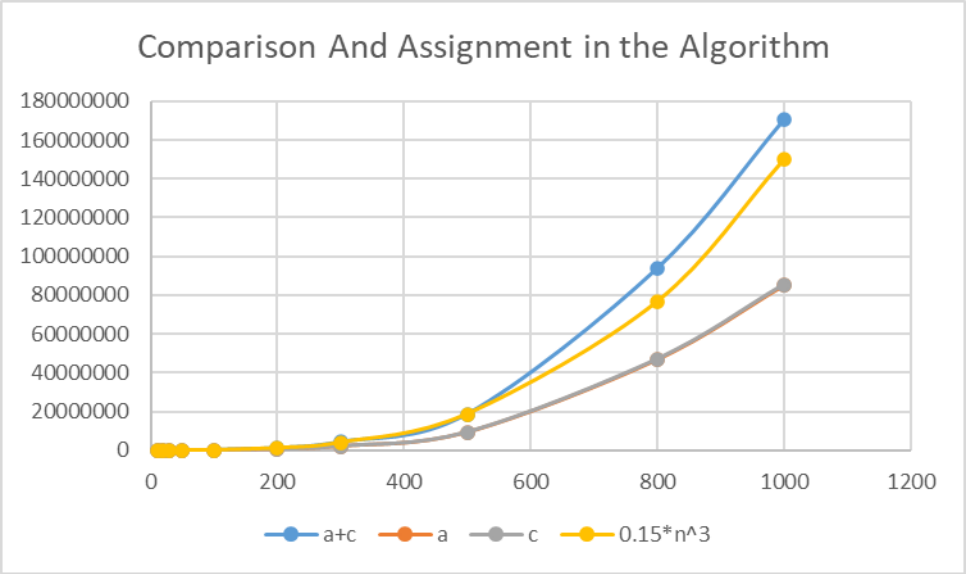
We firstly construct a x' and y' that both length as long as s. So the problem can be reduct to whether s is a interweave of part of x' and part of y', because "part of" x' and y' doesn't limited the repeat times of x and y, so this will cover each possibility of x and y interweave.

And we use a two-dimensional array il [i][j] to represent whether s[i+j] is the interweave of x[i] and y[j], it will be true if the il[i-1][j]==true|| s[i+j]=x[i] or il[i][j-1]==true|| s[i+j]=y[i]

So the problem is reduct into the recursive situation:

$$\text{il [i][j]}\begin{cases} True;\ i=0, j=0 \\ \text{il}[\text{i}-1][\text{j}];\ \text{s}[\text{i}+\text{j}]=\text{x}[\text{i}] \\ \text{il}[\text{i}][\text{j}-1];\ \text{s}[\text{i}+\text{j}]=\text{y}[\text{i} \end{cases}$$

[50 points] Implement your algorithm and test its run time to verify your complexity analysis. Remember that CPU time is not a valid measure for testing run time. You must use something such as the number of comparisons.



To verify the complexity, we use |s|= {10,20,30,50,100,200,300,500,800,1000}, and each s is generate by the random generator, and to eliminate the influence of x and y's content, we assign x=11, y=00. Use "a" to represent assignment operation in the algorithm, "c" to represent comparison operation. "a+c" represent the total assignment and comparison operation.

And we can see that in the figure, the increase of a, c, and a+c suits $0.15*n^3$ very well, so it verified the time complexity of my algorithm.

To verified the length of x and y does not influence the time complexity, I use the |s|=1000, and different x^k and y^k, k={2,10,20,30,50}, to represent x and y in the input(which means different length of 1 repeat or 0 repeat), the assignment and comparison operation count is shown below, we can see that the length of x and y does not influence the time complexity.