به نام خدا

طراحی الگوریتم (بهار ۱۴۰۱)

کوئیز دوم

تاریخ امتحان: ۱۴۰۰/۱۲/۲۱

مدت امتحان: ۲۰ دقیقه

Question: You are given the `EditDistance(X,Y)` function to calculate the minimum edit distance between strings X and Y using insertion, deletion, and substitution. Write a piece of pseudocode to print the steps needed to edit X in order to obtain Y. Determine the time complexity of your solution. Make sure the printed edits can be applied to X from left to right.

```
EditDistance(X, Y) {
    m, n = X.length, Y.length
    for i = 1 to m {
        d[i, 0] = i
    }
    for j = 1 to n {
        d[0, j] = j
    }
    for i = 1 to m {
        for j = 1 to n {
            if X[i-1] = Y[j-1] {
                d[i,j] = d[i-1, j-1]
                ptr[i, j] = 2    // Assume that array indices are zero based.
            } else {
                d[i,j] = min(d[i-1,j], d[i,j-1], d[i-1, j-1]) + 1
                ptr[i, j] = argmin(d[i-1, j], d[i, j-1], d[i-1, j-1])
            }
        }
    }
    return d[m,n], ptr
}
```

Example:

`PrintEditDistance("kitten", "sitting")`

Expected output:

```
substitute "s" for "k" at location 0
substitute "i" for "e" at location 4
insert "g" at location 6
```

Output explanation:

1. **k**itten → **s**itten
2. sitt**e**n → sitt**i**n
3. sittin → sittin**g**

پاسخ: (۸۰ نمره شبهه کد، ۲۰ نمره تحلیل پیچیدگی زمان اجرا)

```
PrintEditDistance(X, Y, ptr, i, j) {
    if i == 0 AND j == 0 {
        return 0
    }
    if i > 0 AND j == 0 {
        l = PrintEditDistance(X, Y, ptr, i-1, j)
        print "delete " + X[i-1] + " at location " + l
        return l
    } else if i == 0 AND j > 0 {
        l = PrintEditDistance(X, Y, ptr, i, j-1)
        print "insert " + Y[j-1] + " at location " + l
        return l+1
    } else if X[i-1] == Y[j-1] {
        l = PrintEditDistance(X, Y, ptr, i-1, j-1)
        return l+1
    } else if ptr[i][j] == 2 {      // substitute
        l = PrintEditDistance(X, Y, ptr, i-1, j-1)
        print "substitute " + X[i-1] + " for " + Y[j-1] + " at location " + l
        return l+1
    } else if ptr[i][j] == 1 {      // insert
        l = PrintEditDistance(X, Y, ptr, i, j-1)
        print "insert " + Y[j-1] + " at location " + l
        return l+1
    } else {                        // delete
        l = PrintEditDistance(X, Y, ptr, i-1, j)
        print "delete " + X[i-1] + " at location " + l
        return l
    }
}

PrintEditDistance(X, Y) {
    _d, ptr = EditDistance(X, Y)
    PrintEditDistance(X, Y, ptr, X.length, Y.length)
}
```

پیچیدگی زمان اجرای الگوریتم، $O(m \times n)$ می‌باشد که m و n به ترتیب طول رشته‌های X و Y می‌باشند.