

I. Edit distance revisited

- (1) Think **recursively**, namely, how to represent global optimality in terms of subproblems
- (2) Take all cases into consideration, sometimes very hard though
- (3) Come up with a **formula**
- (4) What are the initialization values
- (5) Program in a **bottom-up** fashion

II. Edit distance implementation in Python3

```
# A DP program to solve edit distance problem
def editDistDP(x, y):
    m = len(x);
    n = len(y);
    # Create an e-table to store results of subproblems
    e = [[0 for j in range(n + 1)] for i in range(m + 1)]

    # Fill in e[][] in bottom up manner
    for i in range(m + 1):
        for j in range(n + 1):
            # Initialization
            if i == 0:
                e[i][j] = j
            elif j == 0:
                e[i][j] = i
            elif x[i-1] == y[j-1]:
                e[i][j] = min(1 + e[i-1][j], 1 + e[i][j-1], e[i-1][j-1])
            else:
                e[i][j] = 1 + min(e[i-1][j], e[i][j-1], e[i-1][j-1])

    return e[m][n]

# Test case 1
# x = "snowy"
# y = "sunny"

# Test case 2
x = "heroically"
y = "scholarly"

print(editDistDP(x, y))
```

Discussion 7

ECS 122A
May 15, 2019

III. Variant: DNA matching (Homework 5 Problem 8)