14.2.4
Dynamic programming algorithm for edit-distance
As part of the input...

The cost of aligning a character against another character

$\Sigma$: Alphabet

We are given a **cost** function (in a table):

\[
\forall b, c \in \Sigma \quad COST[b][c] = \text{cost of aligning } b \text{ with } c.
\]

\[
\forall b \in \Sigma \quad COST[b][b] = 0
\]

$\delta$: price of deletion of insertion of a single character
Memoizing the Recursive Algorithm (Explicit Memoization)

Input: Two strings
\(A[1 \ldots m]\)
\(B[1 \ldots n]\)

```
EditDistance(A, B)
    int M[0..m][0..n]
    ∀i, j M[i][j] ← ∞
    return edEMI(m, n)
```

```
edEMI(i, j) // A[1 \ldots i], B[1 \ldots j]
    if M[i][j] < ∞
        return M[i][j] // stored value
    if i = 0 or j = 0
        M[i][j] = (i + j)δ
        return M[i][j]
    m1 = δ + edEMI(i − 1, j)
    m2 = δ + edEMI(i, j − 1)
    m3 = COST[A[i]][B[j]]
    + edEMI(i − 1, j − 1)
    M[i][j] = \text{min}(m1, m2, m3)
    return M[i][j]
```
Dynamic program for edit distance

Removing Recursion to obtain Iterative Algorithm

\[
EDIST(A[1..m], B[1..n])
\]

\[
\begin{array}{l}
\text{int } M[0..m][0..n] \\
\text{for } i = 1 \text{ to } m \text{ do } M[i, 0] = i \delta \\
\text{for } j = 1 \text{ to } n \text{ do } M[0, j] = j \delta \\
\text{for } i = 1 \text{ to } m \text{ do} \\
\quad \text{for } j = 1 \text{ to } n \text{ do} \\
\quad \quad M[i][j] = \min \left\{ \begin{array}{l}
COST[A[i]][B[j]] + M[i - 1][j - 1], \\
\delta + M[i - 1][j], \\
\delta + M[i][j - 1]
\end{array} \right. \\
\end{array}
\]

Analysis

Running time is \(O(mn)\).
Dynamic program for edit distance
Removing Recursion to obtain Iterative Algorithm

\[
EDIST(A[1..m], B[1..n])
\]

\[
\text{int } M[0..m][0..n]
\]

for \(i = 1\) to \(m\) do \(M[i, 0] = i\delta\)

for \(j = 1\) to \(n\) do \(M[0, j] = j\delta\)

for \(i = 1\) to \(m\) do

for \(j = 1\) to \(n\) do

\[
M[i][j] = \min\left\{\begin{array}{l}
COST[A[i]][B[j]] + M[i - 1][j - 1], \\
\delta + M[i - 1][j], \\
\delta + M[i][j - 1]
\end{array}\right\}
\]

Analysis

Running time is \(O(mn)\).
Dynamic program for edit distance
Removing Recursion to obtain Iterative Algorithm

$$EDIST(A[1..m], B[1..n])$$

```
int M[0..m][0..n]
for i = 1 to m do M[i, 0] = i\delta
for j = 1 to n do M[0, j] = j\delta

for i = 1 to m do
  for j = 1 to n do
    M[i][j] = \min \left\{ \text{COST}(A[i], B[j]) + M[i-1][j-1], \delta + M[i-1][j], \delta + M[i][j-1] \right\}
```

Analysis

1. Running time is $O(mn)$.
2. Space used is $O(mn)$. 

THE END

... (for now)