# Introduction to Bioinformatics (autumn 2005)

## Excercise 3

| Group | time | place |
|---|---|---|
| Riikka Kaven | Tuesday 18.10 at 12.15–14.00 | BK106 |

1. (Problem 6.4 in J & P) Modify `DPChange` to return not only the smallest *number* of coins but also the correct combination of coins.

2. (Problem 6.6) Give an algorithm to compute the number of different paths from *source* $(0,0)$ to *sink* $(m,n)$ in an $m \times n$ rectangular grid.

3. Find the length of the *longest common subsequence* $LCS(X,Y)$ of $X =$ `stockholm` and $Y =$ `tukholma` (using dynamic programming recurrence at page 175). Visualize the corresponding alignment.

4. Edit distance $d_{\mathtt{ID}}(X,Y)$ gives the minimum number of character insertions and deletions needed to convert $X = x_1x_2\cdots x_m$ into $Y = y_1y_2\cdots y_n$. As mentioned in the lecture, this is the dual of LCS in the sense that $d_{\mathtt{ID}}(X,Y) = |X| + |Y| - 2 * |LCS(X,Y)|$. A direct recursion to compute $d_{\mathtt{ID}}(X,Y)$ is as follows:

   $d(i,j) = \min\{d(i-1,j)+1, d(i,j-1)+1,$ **if** $x_i = y_j$ **then** $d(i-1,j-1)$ **else** $\infty\}$.

   a) How should values $d(i,j)$ be initialized for the first column and first row of the dynamic programming matrix in order the recursion to work? Which value $d(i,j)$ corresponds to $d_{\mathtt{ID}}(X,Y)$?

   b) A more commonly used edit distance, called *Levenshtein distance*, $d_{\mathtt{L}}(X,Y)$ gives the minimum number of insertions, deletions, and *substitutions* to convert $X$ into $Y$. Substitution replaces a character $x_i$ with a character $y_j$. Modify the above recurrence to compute this distance.

5. (Problem 6.32) A string $X$ is called *supersequence* of a string $V$ if $V$ is a subsequence of $X$. For example, `ABLUE` is a supersequence for `BLUE` and `ABLE`. Given strings $V$ and $W$, devise an algorithm to find the shortest supersequence common to $V$ and $W$.