CS 253 Programming Assignment 3

Fall 2003 (Due 10/23/2003) at the beginning of class

Be sure to read the Programming Style document when turning in your assignment. You must work in a small group for this assignment with different members than the last two assignments.

Do problem 15-3-a, the edit distance problem. Provide both a recursive and iterative solution to this. Develop invariants that describe the correctness implement as much as possible. Also, validate your run-time complexity experimentally.

A recursive solution to the edit distance problem looks similar to that of the LCS problem and is as follows:

Define c(i,j) as the cost of transforming string x[i..m] into y[j..n] using

If x[i]=y[j]

then c(i,j) = c(i+1,j+1) + cost_copy(x[i],y[i])

```
else c(i,j) = min {     c(i+1,j+1) + cost_replace(x[i],y[j]),

                        c(i+1,j) + cost_delete(x[i],-)   where “-“ means leave untouched

                        c(i,j+1) + cost_insert(-,y[j])

                        c(i+2,j+2) + cost_twiddle((x[i],x[i+1],y[j],y[j+1]) (only if i and j <= m-1 and n-1)

                        c(m+1,j) + cost_kill(x[i]..x[m],-)    }
```

Boundary Cases

```
c(m+1,n+1) = 0          c(i,n+1) = c(i+1,n+1) + cost_delete(x[i],-)

                        c(m+1,j) = c(m+1,j+1)  + cost(insert(-,y[j])
```

The initial call is made via c(1,1).

An iterative solution builds an (m+1 x n+1) distance matrix C in the same fashion as the LCS problem. The bottom right corner of the distance matrix contains the minimal cost. The solution is reconstructed similar to LCS.

You will need to determine appropriate values for each cost based on the “readability” of the resulting transformations (try this on text).

For extra credit do problem 15-3-b.