CS3383, Winter 2019 Assignment # 4 Sample solutions

Rasoul Shahsavarifar
Faculty of Computer Science, UNB

Questions 1) Basic idea: add "cuts" in position 1 and n, so we can compute the cost of further cutting a string defined by the cut at its start and the cut at its end.

$StringCut(length\ \ n, count\ \ m, array\ \ C[1..m]) : returns\ \ integer$
Let $M[0..m+1, 0..m+1]$ be a 2d array of integers
Extend C to position 0 and $m+1$
$C[0] \leftarrow 1$
$C[m+1] \leftarrow n$
$for\ \ i\ \ from\ \ 0\ \ to\ \ m$
    $M[i, i+1] \leftarrow 0$
$for\ \ h\ \ from\ \ 2\ \ to\ \ m+1$
    $for\ \ i\ \ from\ \ 0\ \ to\ \ m+1-h$
        $j \leftarrow i+h$
        $M[i,j] \leftarrow M[i,i+1] + M[i+1,j] + (C[j] - C[i] + 1)$
        $for\ \ k\ \ from\ \ i+2\ \ to\ \ j-1$
            $if\ \ M[i,j] > M[i,k] + M[k,j] + (C[j] - C[i] + 1)$
                $M[i,j] \leftarrow M[i,k] + M[k,j] + (C[j] - C[i] + 1)$
$return\ \ M[0, m+1]$

Note for TA: Getting the indices right is tricky; look for a logical approach, through the execution might be a bit off. Students may have dealt with the overall string differently (if they didn't add the "cuts" to reflect first and last).
Student might also have tried basing the computation on n, but there will be difficulties due to many of the positions in the string not being cuts. This approach is also less efficient.

Question 2) a)The pseudocode for this part is as follows:

$Postage(n, \{d_i\}, \{s_i\}, P)$
Let M be a 2-dimensional table of numbers,$(n+1) \times (P+1)$
$M[0,0] \leftarrow 0$
$for\ \ j\ \ from\ \ 1\ \ to\ \ P$
    $M[0,j] \leftarrow \infty$ //no stamp but still need postage
$for\ \ i\ \ from\ \ 1\ \ to\ \ n$
    $M[i,0] \leftarrow 0$ //no postage left to make
    $for\ \ j\ \ from\ \ 1\ \ to\ \ P$
        $M[i,j] \leftarrow M[i-1,j]$
        $if\ \ d_i \leq j\ \ \&\ \ M[i,j] > M[i,j-d_i] + s_i$
            $M[i,j] \leftarrow M[i,j-d] + s_i$
$return\ \ M[n,P]$

b) The pseudocode and the trace function for this part are as follows:

$Postage(n, \{d_i\}, \{s_i\}, P)$
Let M be a 2-dimensional table of numbers,$(n+1) \times (P+1)$
Let T be a 2-dimensional table of labels,$(n+1) \times (P+1)$
Let D be a 1-dimensional array of n integers
$M[0,0] \leftarrow 0$
$for\ \ j\ \ from\ \ 1\ \ to\ \ P$
    $M[0,j] \leftarrow \infty$ //no stamp but still need postage
$for\ \ i\ \ from\ \ 1\ \ to\ \ n$
    $M[i,0] \leftarrow 0$ //no postage left to make
    $for\ \ j\ \ from\ \ 1\ \ to\ \ P$
        $M[i,j] \leftarrow M[i-1,j]$
        $T[i,j] \leftarrow "skip"$
        $if\ \ d_i \leq j\ \ \&\ \ M[i,j] > M[i,j-d_i] + s_i$
            $M[i,j] \leftarrow M[i,j-d] + s_i$
            $T[i,j] \leftarrow "use"$
$if\ \ M[n,P] = \infty$
    print "postage not possible"
$else$
    $for\ \ i\ \ from 1\ \ to\ \ n$
        $D[i] \leftarrow 0$
    $Trace(T, \{d_i\}, D, n, P)$
$return\ \ M[n,P]$

The $Trace$ function:

$$
\begin{array}{l}
Trace(T, \{d_i\}, D, i, j) \\
if\ \ j > 0 \\
\qquad if\ \ T[i, j] = \text{"}skip\text{"} \\
\qquad\qquad Trace(T, \{d_i\}, D, i-1, j) \\
\qquad else \\
\qquad\qquad D[i] \leftarrow D[i] + 1 \\
\qquad\qquad Trace(T, \{d_i\}, D, i, j-d) \\
\qquad\qquad \text{print "use } d_i \text{ stamp"}
\end{array}
$$

c) The asymptotic running time is $\Theta(nP)$

Question 3) a)

$$
\begin{array}{l}
Trip(arrya[1..n], lenght\ \ n) : returns\ \ sequence\ \ S \\
\text{Let } \ M[1..n] \text{be an array of numbers} \\
\text{Let } \ T[1..n] \text{be an array of integers} \\
for\ \ i\ \ from\ \ 1\ \ to\ \ n \\
\qquad M[i] \leftarrow (200 - a[i])^2 \\
\qquad T[i] \leftarrow 0 \\
\qquad for\ \ j\ \ from\ \ i-1\ \ down\ \ to\ \ 1 \\
\qquad\qquad if\ \ M[i] > M[j] + (200 - (a[i] - a[j]))^2 \\
\qquad\qquad\quad M[i] \leftarrow M[j] + (200 - (a[i] - a[j]))^2 \\
\qquad\qquad\quad T[i] \leftarrow j \\
return\ \ Trace(T, n)
\end{array}
$$

where $Trace$ is as follows:

$$
\begin{array}{l}
Trace(arryT[1..n], index\ \ i) : returns\ \ sequence\ \ S \\
\text{Let } S \text{ be an empty array} \\
if\ \ i > 0 \\
\qquad S \leftarrow Trace(T, T[i]) \\
\qquad append\ \ i\ \ to\ \ S \\
return\ \ S
\end{array}
$$

b) The analysis is straightforward, and the running time is $\Theta(n^2)$. Is the algorithm a pseudo-polynomial algorithm?