# Programming Assignment 3: Report

## (a) [10 points] The recursive relation that you need to use to compute maximizeSentimentalValue().

maximizeSentimentalValue(i, w) is short as mSV(i, w) that tracks the maximum sentimental value with current items and weight.

$$mSV(i,w) = \begin{cases} 0, & \text{if } i = 0. \\ \text{mSV(i - 1, w)}, & \text{if } w_i > w. \\ \max\{mSV(i-1,w), v_i + mSV(i-1, w-w_i)\} & \text{otherwise.} \end{cases} \tag{1}$$

## (b) [30 points] The recursive relation that you need to use to compute cleverWidthReduction().

The "content" value of each pixel is stored in a matrix contents. i index is the number of row from the top, and j index is the number of column from left. The "content" value of pixel at i-th row, j-th column is contents[i][j].

The function calculating the minimum sum of content along a path from the 1-th row to the h-th row is named minContentSum(i, j), where i and j are the height and width of the pixel that we are ending at. The valid j range of contents and range of matrix storing function minContentSum(i, j) result is from 1 to w, and the valid i range is from 1 to h. Function minContentSum(i, j) is short as mCS(i, j) below.

Each element in the dynamic programming result matrix DP[i][j] is computed by mCS(i, j). Then we know the pixel where the vertical path ends is at the h-th row with the minimum sum of content. Tracking back from that pixel gives us a vertical path to delete. Then, delete and update $(width - desired\_width)$ times to get the desired width.

$$mCS(i,j) = \begin{cases} contents[i][j], & \text{if } i = 1 \text{ and } 1 \le j \le w. \\ \infty, & \text{if } j < 1 \text{ or } j > w. \\ contents[i][j] + \min \begin{cases} mCS\text{(i - 1, j - 1)}, \\ mCS\text{(i - 1, j)}, \\ mCS\text{(i - 1, j + 1)}, \end{cases} & \text{if } 1 < i \le h \text{ and } 1 \le j \le w. \end{cases} \tag{2}$$