# CSE101 HW7

Yuhang Jiang

June 7, 2023

# 1.

1. Define the subproblems:
   $dp[i][j][k]$: The maximum value of only consider the first $i$ groups, and there are $j$ people in the upper and $k$ people in the lower.

2. Define and evaluate the base cases:
   $dp[0][0][0] = 0$. Only consider the first 0 groups, the value is 0, and there are 0 people in the upper and 0 people in the lower.

3. Establish the recurrence for the tabulation.

   - Case 1: Put the current group on the upper if there is space on the upper.
     $dp[i][j][k] = dp[i-1][j-p[i]][k] + v[i]$
   - Case 2: Put the current group on the lower if there is space on the lower.
     $dp[i][j][k] = dp[i-1][j][k-p[i]] + v[i]$
   - Case 3: Ignore current group. $dp[i][j][k] = dp[i-1][j][k]$

4. Determine the order of subproblems:
   Order the subproblems from 0 to $n$.

5. Final form of output:
   $\min\{dp[n]\}$, the maximum value of considering all groups, regardless of how many people in the upper and lower.

6. Put it all together as pseudocode:

   $dp[0][0][0] = 0$, otherwise $dp[i][j][k] = -\infty$
   **for** $i = 1...n$ **do**
   **for** $j = 0...U$ **do**
   **for** $k = 0...L$ **do**
   **if** $j \geq p[i]$ **then**
   $dp[i][j][k] = \max(dp[i][j][k], dp[i-1][j-p[i]][k] + v[i])$
   **if** $k \geq p[i]$ **then**
   $dp[i][j][k] = \max(dp[i][j][k], dp[i-1][j][k-p[i]] + v[i])$
   $dp[i][j][k] = \max(dp[i][j][k], dp[i-1][j][k])$
   **return** $\min dp[n][0...U][0...L]$

7. Runtime analysis:
   The code contains a triple loop, the internal complexity of the triple loop is $O(1)$. The first loop is executed $n$ times, the second loop is executed $U+1$ times, and the third loop is executed $L+1$ times.
   Total time: $O(nUL)$

## 2.

1. Define the subproblems:
   $dp[i][j]$ is the minimum height of a bookshelf: contains first $i$ books, and the last layer of shelf has $j$ books.

2. Define and evaluate the base cases:
   $dp[0][0] = 0$, otherwise $dp[i][j] = \infty$. 0 book has only one placement, last layer of shelf has 0 book.

3. Establish the recurrence for the tabulation.
   Let $\text{maxh}(l, r)$ denotes the maximum height of all books from $l$ to $r$.
   Let $\text{sumw}(l, r)$ denotes the sum width of all books from $l$ to $r$.
   Special case: If $l > r$, maxh=sumw=0, it means there are 0 books in the interval.

   - Case 1: Put the next book $(i + 1)$ on the current shelf. Restriction: the sum width of the current shelf and next book $\leq W$
     $dp[i + 1][j + 1] = dp[i][j] + max(h[i + 1] - \text{maxh}[i - j + 1][i], 0)$
     $(\text{sumw}[i - j + 1][i] + w[i + 1] \leq W)$

   - Case 2: Place the next book on a new shelf.
     $dp[i + 1][1] = dp[i][j] + h[i + 1]$

4. Determine the order of subproblems:
   Order the subproblems from 0 to $n$.

5. Final form of output:
   $\min\{dp[n]\}$, the minimum height of the bookshelf contains all books, regardless how many books in the last layer of shelf.

   preprocessing $\text{maxh}[l][r]$ and $\text{sumw}[l][r]$
   if $l > r$ then $\text{maxh}[l][r] = \text{sumw}[l][r] = 0$
   $dp[0][0] = 0$, otherwise $dp[i][j] = \infty$
   **for** $i = 0...n - 1$ **do**
   **for** $j = 0...i$ **do**
   **if** $\text{sumw}[i - j + 1][i] > W$ **then** break
   **if** $dp[i][j] == \infty$ **then** continue
   **if** $\text{sumw}[i - j + 1][i] + w[i + 1] \leq W$ **then**
   $dp[i+1][j+1] = \min(dp[i+1][j+1], dp[i][j] + \max(h[i+1] - \text{maxh}[i - j + 1][i], 0))$
   $dp[i + 1][i] = \min(dp[i + 1][1], dp[i][j] + h[i + 1])$
   **return** $\min\{dp[n][0...n]\}$

6. Runtime analysis:

The code contains a double loop, if preprocessing maxh and sumw, the internal complexity of the double loop is $O(1)$.
The first loop is executed $n + 1$ times, the second loop is executed up to $n + 1$ times.
Time complexity: $O(n^2)$