6.19 Coin denominations $x_1, x_2, \ldots, x_n$

For each denomination there is an unlimited supply of coins.

We need to make change for a value $v$ using at most $k$ coins. Therefore, we can assume that we have total of $k \cdot n$ coins.

Let $y_1, y_2, \ldots, y_{kn}$ be the coins.

Let $C(t, j)$ be the optimal # of coins from $y_1, y_2, \ldots, y_j$ to change $t$, where $1 \leq t \leq v$ and $1 \leq j \leq kn$. Let $u_j$ be the value of coin $y_j$.

The # of subproblems is $O(knv)$

We can write $C(t, j)$ recursively as

$$C(t, j) = \min \{ C(t - u_j, j - 1) + 1, C(t, j - 1) \}$$

**Basis** $C(0, j) = 0$ for any $a \leq 0$

and $C(t, 0) = 0$ for any $t \geq 0$

**Memoized** $C(t, j)$

- If $t \leq 0$ return 0;
- If $j = 0$ return 0;
- If $C(t, j)$ is already computed then return $C(t, j)$

else return $C(t, j) = \min \{ C(t - u_j, j - 1) + 1, C(t, j - 1) \}$
4. Retrieve \( C(t,j) \)
   
   \[
   \begin{cases}
   & \text{if } t < 0 \text{ return null.} \\
   & \text{if } j = 0 \text{ return } 0 \\
   & \text{if } C(t,j) = 1 + C(t-u_j, j-1) \text{ then} \\
   & \quad \text{Print } y_j \\
   \end{cases}
   \]

5. **Iterative solution**

   **Subproblems**

   \[
   \begin{array}{c}
   y_1, y_2, y_3, \ldots, y_j, \ldots, y_{kn} \\
   \end{array}
   \]

   We evaluate row-wise

   For \( t = 1, 2, \ldots, kn \)

   For \( j = 1, 2, \ldots, kn \)

   \[
   C(t,j) = \min \{ 1 + C(t-u_j, j-1), C(t,j) \}
   \]
**Chain Matrix Multiplication**

We need to optimally multiply

\[ A_1 \times A_2 \times \cdots \times A_n \]

where the dimension of \( A_l \) is \( m_{l-1} \times m_l \), \( l = 1, 2, \ldots, n \).

The dimension of the product matrix is \( m_1 \times m_{n+1} \).

Let \( C(i, j) \) be the optimal multiplication cost of the matrices

\[ A_i \times A_{i+1} \times \cdots \times A_j \]

There are \( n^2 \) subproblems.

**Basis:**
- \( C(1, 2) = m_1 m_2 m_3 \)
- \( C(2, 3) = m_2 m_3 m_4 \)
- \( C(n-1, n) = m_{n-1} \cdot m_n \cdot m_{n+1} \)

**Recurrence Formula:**

\[
C(i, j) = \min_{i \leq k < j} \left( C(i, k) + C(k+1, j) + m_i \cdot m_{k+1} \cdot m_{j+1} \right)
\]

Each subproblem requires \( O(n^2) \) time once the smaller subproblem values are known. The memoized cost is \( O(n^3) \).
In order to compute $C(i,j)$, we note that all the values marked $x$ must be evaluated before $C(i,j)$ can be computed. We evaluate row by row from left to right:

For $i = 1, 2, \ldots, n$

For $j = i, i+1, \ldots, n$

$$C(i,j) = \min_{1 \leq k < j} \left( C(i,k) + \min_{k+1 \leq l \leq j} \left( C(k+1,l) + m_i + m_{j+1} \right) \right)$$
6.4 Subproblems: Define $S(i)$ to be 1 if

$s[1, \ldots, i]$ is a sequence of valid words. Otherwise it is 0.

Recurrence formula

$$S(i) = \max_{1 \leq j < i} \left\{ S(j) : \text{dict}(s[j+1, \ldots, i], \text{true}) \right\}$$

If $S(j)$ is false for every 

$$\text{dict}(s[j+1, \ldots, i], \text{true})$$

$S(j)$ is returned 0.

The given input sequence is a sequence of valid words if $S(n)=1$.

The number of subproblems is $n$. Each subproblem can be solved in $O(n^2)$ time. Total work is $O(n^3)$.

The basis is $S(i)$ is true for $i < 0$.

Iterative solution is easy:

$$S(0) = \ldots$$
6.7 \( L(i, j) \): length of the longest palindromic subsequence of string \( S[1 \ldots n] \).

For \( 1 \leq i \leq n \) and \( 1 \leq j \leq n \).

**Basis**
- \( L(i, i) = 1 + i \)
- \( L(i, j) = 0 \) if \( i > j \)

We can show that:

\[
L(i, j) = 1 + L(i + 1, j - 1) \quad \text{if} \quad x_i = x_j \\
= \min \{L(i + 1, j), L(i, j - 1)\} \quad \text{otherwise}
\]

There are \( O(n^2) \) subproblems. Each subproblem takes \( O(1) \) time to compute.

The order with which we compute \( L(i, j) \) is given below:

- For \( j = 2, 3, \ldots, n \)
- For \( i = 1, 2, \ldots, j \)

\[ L(i, j) = \ldots \]
$V^{+}(u) =$ size of minimum vertex cover that includes $u$

$V^{-}(u) =$ size of minimum vertex cover that does not include $u$.

$\therefore V(u) = \min \{ V^{+}(u), V^{-}(u) \}$

Consider:

$V^{+}(u) = 1 + \sum_{\forall c_i} V^{-}(c_i) \quad (\text{all the edges}\ (u, c_i) \text{ are covered by } u)$

$V^{-}(u) = \sum_{\forall c^+} V^{+}(c_i)$

Note that:

$V^{+}(u) = 1$ for all leaf nodes

$V^{-}(u) = 0$