李青航 SA22225226

15.2-1

方案 $(A_1A_2)((A_3A_4)(A_5A_6))$

最小代价$5 \cdot 50 \cdot 6 + 3 \cdot 12 \cdot 5 + 5 \cdot 10 \cdot 3 + 3 \cdot 5 \cdot 6 + 5 \cdot 3 \cdot 6 = 1500 + 180 + 150 + 90 + 90 = 2010$

15.2-2

见算法1

Algorithm 1 MATRIX-CHAIN-MULTIPLY(A, s, i, j)

```
if i == j then
    return A_i
end if
return MATRIX-CHAIN-MULTIPLY(A, s, i, s[i, j]) · MATRIX-CHAIN-MULTIPLY(A, s, s[i, j] + 1, j)
```

15.3-1

穷举快

穷举法根据组合数学知识，时间复杂度是一个指数复杂度。而RECURSIVE-MATRIX-CHAIN也是一个指数复杂度的，但是还有递归函数调用开销，还会重叠算某些子问题（穷举没有重叠）,重复调用函数。

15.3-2

图太难画，略。归并排序并没有重叠子问题，每次归并，没有重复可用的操作，所以备忘技术没有用

15.4-3

见算法2，3

其中数组b, c是引用传递，可以被函数内部改变。

15.5-3

在每次计算$e[i, j]$时都重新计算了$\omega(i, j)$，每次多出$\Theta(j - i)$次加法。

每次多出$O(n)$规模，总的时间复杂度增加到$O(n^3)$

Algorithm 2 MEMO-LCS-LENGTH-AUX(X, Y, c, b)

m=$|X|$
n=$|Y|$
if $c[m, n]! = 0$ or $m == 0$ or $n == 0$ **then**
  return $c[m, n]$
end if
if $x_m == y_n$ **then**
  $b[m, n] = \nwarrow$
  $c[m, n]$ =MEMO-LCS-LENGTH-AUX($X[1 \ldots m-1], Y[1 \ldots n-1], c, b$)+1
else if MEMO-LCS-LENGTH-AUX($X[1 \ldots m-1], Y, c, b$)$\geq$MEMO-LCS-LENGTH-AUX($X, Y[1 \ldots n-1], c, b$) **then**
  $b[m, n] = \uparrow$
  $c[m, n]$ =MEMO-LCS-LENGTH-AUX($X[1 \ldots m-1], Y, c, b$)
else
  $b[m, n] = \leftarrow$
  $c[m, n]$ =MEMO-LCS-LENGTH-AUX($X, Y[1 \ldots n-1], c, b$)
end if
return $c[m, n]$

Algorithm 3 MEMO-LCS-LENGTH(X, Y)

let $b[1 \ldots m, 1 \ldots n]$ and $c[1 \ldots m, 1 \ldots n]$be new table
for $i = 1$ to m **do**
  $c[i, 0] = 0$
end for
for $i = 1$ to n **do**
  $c[0, i] = 0$
end for
MEMO-LCS-LENGTH-AUX(X, Y, c, b)//b and c,passed by reference
return b and c