David Wang
Andrew Nguyen
CSE 100

Homework #3

Exercise 1)

part 1:

```
Matrix-Chain-Multiply(A, s, i, j)
    if j == i
        return A[i]
    if j == (i + 1)
        return A[i] * A[j]
    b = Matrix-Chain-Multiply(A, s, i, s[i][j])
    c = Matrix-Chain-Multiply(A, s, s[i][j] + 1, j)
    Matrix-Multiply(b, c)
```

part 2: (underlined min answer)

$p = \langle 6, 10, 3, 50, 5 \rangle$ (indices 0 1 2 3 4)

$m[i,j] = \begin{cases} 0 & \text{if } i = j \\ \min_{i \le k < j} \{ m[i,k] + m[k+1,j] + p_{i-1} p_k p_j \} \end{cases}$

A1 A2 A3 A4

$m[1,1] = 0$ $m[2,2] = 0$ $m[3,3] = 0$ $m[4,4] = 0$

$m[1,2] = m[1,1] + m[2,2] + p_0 p_1 p_2 = 0 + 0 + 6 \cdot 10 \cdot 3 = 180$ $s[1,2] = 1$

$m[2,3] = m[2,2] + m[3,3] + p_1 p_2 p_3 = 0 + 0 + 10 \cdot 3 \cdot 50 = 1500$ $s[2,3] = 3$

$m[3,4] = m[3,3] + m[4,4] + p_2 p_3 p_4 = 0 + 0 + 3 \cdot 50 \cdot 5 = 750$ $s[3,4] = 3$

$m[1,3] = m[1,1] + m[2,3] + p_0 p_1 p_3 = 0 + 1500 + 6 \cdot 10 \cdot 50 = 4{,}500$ $s[1,3] = 1$

or $m[1,2] + m[3,3] + p_0 p_2 p_3 = 180 + 0 + 6 \cdot 3 \cdot 50 = 1{,}080$, $s[1,3] = 2$

part 2 cont.:

$m[2,4] = A_2 \; m[2,2] + m[3,4] + P_1 P_2 P_4$

$= 0 + 750 + 10 \cdot 3 \cdot 5 = \underline{900} \quad s[2,4] = 2$

or $m[2, {}^{A_2A_3}?] + m[{}^{A_4}4,4] + P_1 P_3 P_4 =$

$= 1500 + 0 + 10 \cdot 50 \cdot 5 = 4000 \quad s[2,4] = 3$

$m[1,4] = A_1 \; (A_2A_3A_4)$

$= m[1,1] + A[2,4] + P_0 P_1 P_4$

$= 0 + 900 \quad 6 \cdot 10 \cdot 5 = 1,200 \quad s[1,4] = 1$

$(A_1A_2)(A_3A_4)$

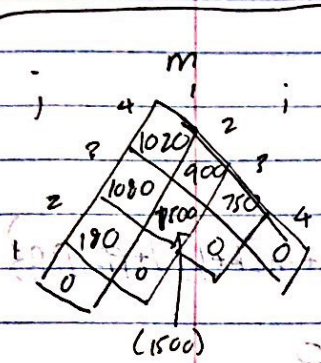
$m[1,2] + m[3,4] + P_0 P_2 P_4 \qquad s[1,4] = 2$

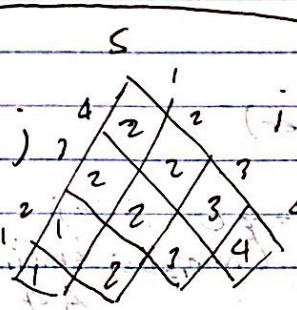
$= 180 + 750 + 6 \cdot 3 \cdot 5 = \underline{1020}$

$(A_1A_2A_3)\; A_4$

$m[1,3] + m[4,4] + * \; P_0 P_3 P_4$

$= 1080 + 0 + * \; 6 \cdot 50 \cdot 5 = 2,580 \quad s[1,4] = 3$





$A_1 A_2 A_3 A_4$

$m[1,4] = 1020$

$s[1,4] = 2$

A_1A_2 $m[1,2]$ $s[1,2] = 1$ → A_1, A_2

A_3A_4 $m[3,4]$ $s[3,4] = 3$ → A_3, A_4

optimal parenthesis $(A_1A_2)(A_3A_4)$

$A_1 = (6 \times 10) \quad A_3 = (3 \times 50)$

$A_2 = (10 \times 3) \quad A_4 = (50 \times 5)$

$= ((6 \times 10)(10 \times 3))((3 \times 50)(50 \times 5))$

Ex 1)

Part 3:

Using the equation

$$M_{i,j} = \min_{i \le k < j} \{ M_{i,k} + M_{k+1,j} + P_i P_{k+1} P_{j+1} \}$$

When we parenthesize $A_1 \cdot A_2 \cdot A_3 \cdots A_n$ it is the same as knowing the optimal parenthesization of $(A_1 \cdot A_2 \cdots A_i)$ and $(A_{i+1} \cdots A_n)$

Thus the subproblems overlap, since an optimal solution depends on optimal subproblems. Thus, using a greedy algorithm would not work.

Exercise 2

1) Brute Force $O(2^n)$: Since it is a brute force approach using binary search trees, it doubles per each step of the tree

3)
```
Memoized-cut-rod(p,n)
  let r[0..n] be new int[] = new r[n]
  for (int i=0; i<n; i++){
    r[i] = -∞
  return memoized-cut-rod-aux(p, n, r)
```

$O(n^2)$

```
memoized-cut-rod-aux(p, n, r)
if(r[n] >= 0 {
      return r[n]
} if (n==0) {
    q = 0
} else {
   q = -∞
   for (int i=1; i<=n; i++){
       q = max(q, p[i] + memoized-cut-rod-aux(p, n-i,
   }
  r[n] = q
  return q
```

4.) [redacted]

```
Bottom-Up-Cut-Rod(p, n)
    r[n]
    r[0] = 0
    for(for j = 1 to n)
        q = -∞
        for(i = 1 to j)
            q = max(q, p[i] + [j - i])
        r[j] = q
    return r[n]
```

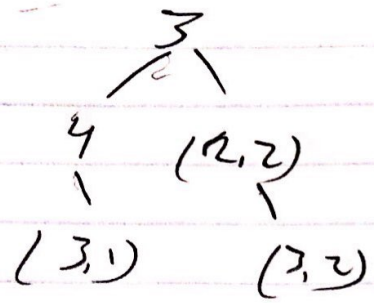
Order: $O(n^2)$

5)

Item	Weight	Price
	5	$6
	6	$11 // Doesn't fit
	3, 2	$5 + $3 = $8
	3, 2	$5 + $2 = $7
	4	$7
	2, 2	$2 + $3 = $5
	2, 1	$2
	2	$3

Item Combination 3 & 2 would be the optimal w/ $8, capacity 5.

6)



7) The top-down method would be faster because it is a recursive function solving less repeated problems, remembering what is already computed

8) Item 5 is the max value in the list, therefore it would be the first element taken in. Since the weight is > than 5, the algorithm goes down the list to item 3 and stores it instead, since the weight does not exceed 5. At the end of all the sorting, only 3 remains.

Exercise 3

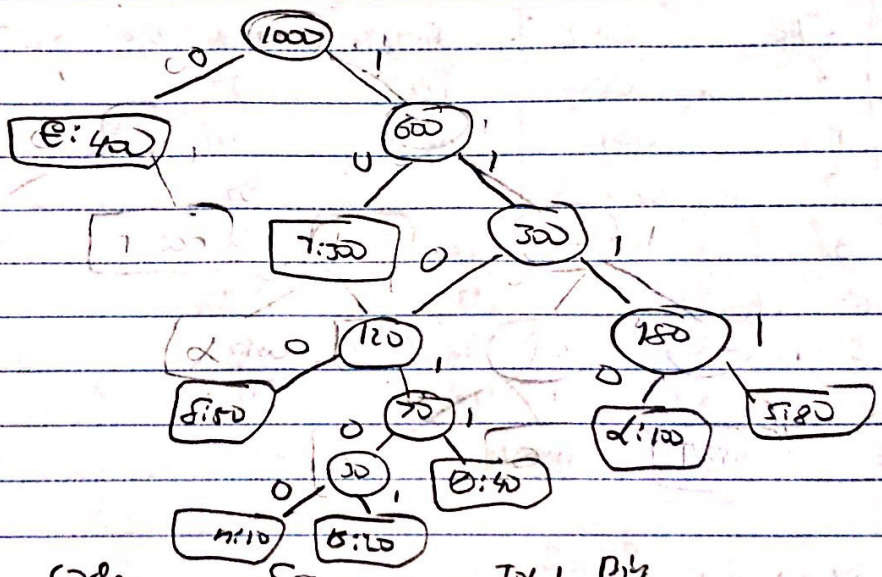
1)

Character	Code	Freq	Total Bits = Freq × Bit length
α	000	100	300
β	001	20	60
γ	010	300	900
δ	011	50	150
ε	100	400	1200
ζ	101	80	240
η	110	10	30
θ	111	40	120

Total Bits: 3,000

Fixed-length

2)



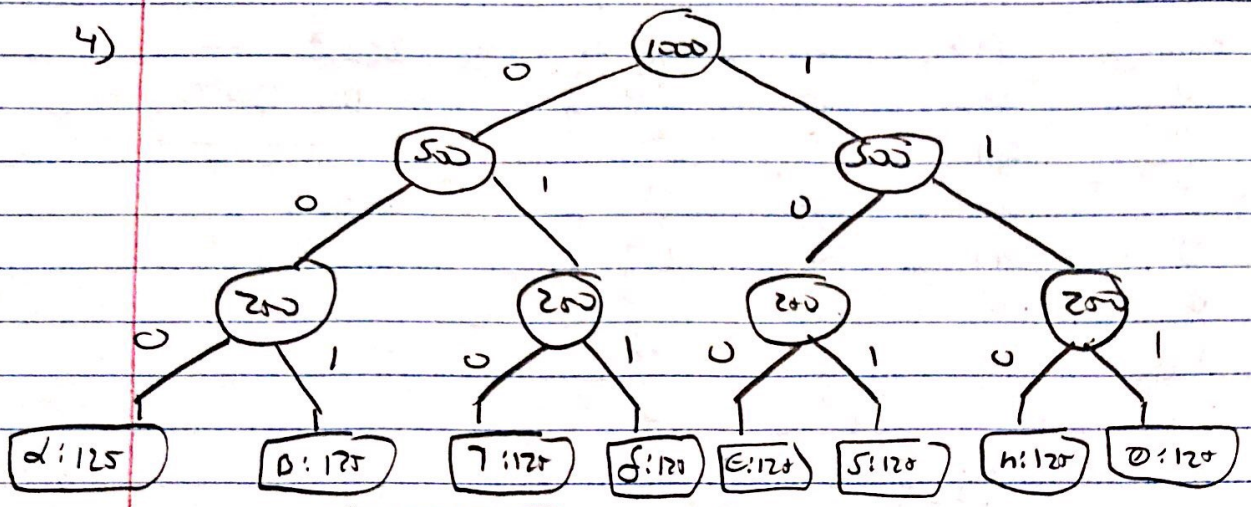
Char	Code	Freq	Total Bits
α	111	100	400
β	111101	20	120
γ	10	300	600
δ	1100	50	200
ε	0	400	400
ζ	1111	80	320
η	110100	10	60
θ	11011	40	200

3)

Total Bits: 2,300

Variable length

4)



Char	Code	Freq	Tot. # bits
α	000	120	360
β	001	120	360
γ	010	120	360
δ	011	120	360
ε	100	120	360
ζ	101	120	360
η	110	120	360
θ	111	120	360

5) Total size = 3000 bits

6) The tree will be equal in size (left & right branches).
Code Bits will generally be equal (in our case 3).
All the characters will be at the end of the tree (= leaf).