CSE 100: Algorithm Design and Analysis
Midterm 3

Spring 2019

Note: Please write down your name on every page. The maximum score you can earn is 100 points. Your score will be capped at 100 if it exceeds 100.

When your answer is incorrect, you will likely get more partial points if you showed intermediate steps. You may run out of time, so please use your time wisely.

Problem	Points earned	Max
1		10
2		15
3		15
4		15
5		15
6		15
7		15
8		40
min {Sum, 100}		100

Name _____

1. (10 points) In the rod cutting problem, we are given as input, $p_0, p_1, p_2, ..., p_n$, where p_i denotes the price of a rod/piece of length i. We are interested in cutting a given rod of length n into pieces of integer lengths in order to maximize the revenue; here we are only interested in finding the maximum revenue. Cutting is free. Let r_i denote the maximum revenue one can get out of a rod of length i. Fill out the following DP table. If your answer is correct, you will get full points. Otherwise, you will get partial points only when you show intermediate calculations using recursion.

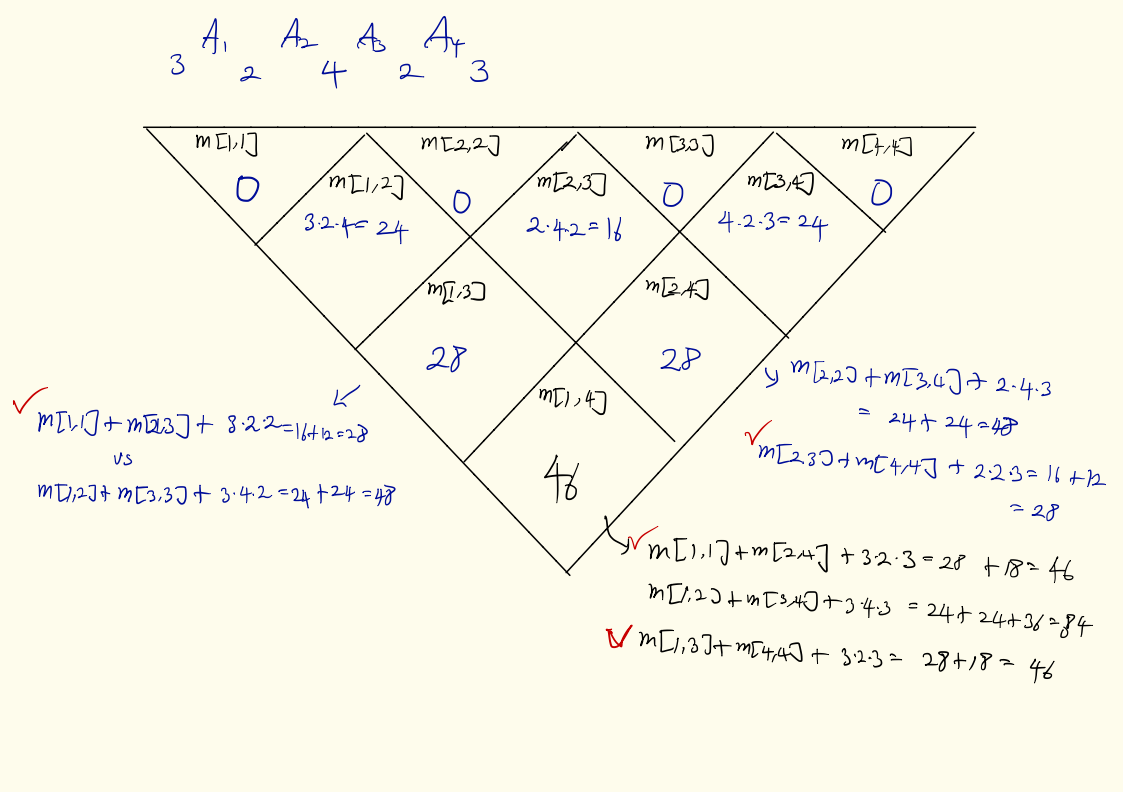
i	1	2	3	4	5	6
p_i	1	5	4	8	9	10
r_i						

Sol.

i	1	2	3	4	5	6
p_i	1	5	4	8	9	10
r_i	1	5	6	10	11	15

2. (15 points) We would like to minimize the number of (scalar) multiplications used to compute the product of four matrices, $A_1A_2A_3A_4$, where the matrices have dimensions $3 \times 2, 2 \times 4, 4 \times 2, 2 \times 3$, respectively. Recall that we defined $m[i, j]$ to the be the minimum number of multiplications needed to compute the product $A_iA_{i+1} \cdots A_j$. Fill out the DP table completely. In other words, compute $m[1, 1], m[2, 2], m[3, 3], m[4, 4]$, $m[1, 2], m[2, 3], m[3, 4]$, $m[1, 3], m[2, 4]$, $m[1, 4]$ using the recursion we learned.
(Note: $m[1, 1], m[2, 2], m[3, 3], m[4, 4]$ each are worth 0.5pt; $m[1, 2], m[2, 3], m[3, 4]$ each are worth 1pt; $m[1, 3], m[2, 4]$ each are worth 3pts; and $m[1, 4]$ is worth 4pts.)

Sol.



3. (15 points) In the LCS problem, we are given as input two sequences, $X = \langle x_1, x_2, \ldots, x_m \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$ and would like to find a longest subsequence common to both. Towards this end, we defined $c[i, j]$ to be the length of LCS of X_i and Y_j, where $X_i := \langle x_1, x_2, ...x_i \rangle$ and $Y_j := \langle y_1, y_2, ...y_j \rangle$.

   (a) (5 points) Give a recursion for computing $c[i, j]$. Do not forget the base cases.
   Sol. See the lecture slides or the textbook

   (b) (10 points) Fill out the the following empty LCS DP table of entries $c[i, j]$.

i	j	0	1	2	3	4
		y_j	A	B	B	A
0	x_i					
1	A					
2	C					
3	B					
4	A					
5	B					

   Sol.

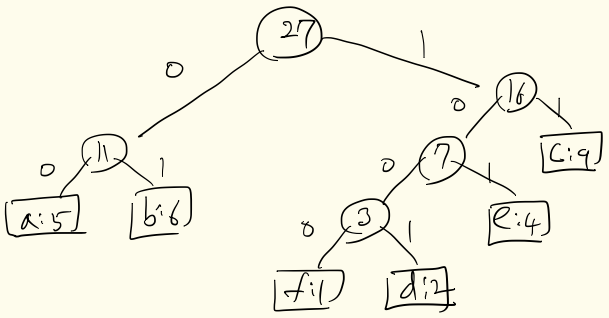
i	j	0	1	2	3	4
		y_j	A	B	B	A
0	x_i	0	0	0	0	0
1	A	0	1	1	1	1
2	C	0	1	1	1	1
3	B	0	1	2	2	2
4	A	0	1	2	2	3
5	B	0	1	2	3	3

4. (15 points) Huffman Code. Suppose we have a text consisting only of a, b, c, d, e, f where each character appears with the following frequency:

a	b	c	d	e	f
5	6	9	2	4	1

   (a) (11 points) Show the code built by the Huffman algorithm, *both as a tree and as a list* (character, codeword). When combining two trees, *the tree with lowest root frequency becomes the left child* and the tree with the second-lowest root frequency becomes the right child. Left children are associated with the bit 0, right children with the bit 1.

   Sol.



a: 00

b: 01

c: 11

d: 1001

e: 101

f: 1000

(b) (2 points) How many bits are required to represent the input using this Huffman code? **Sol.** 5*2+6*2+9*2+2*4+4*3+1*4=64

(c) (2 points) How many bits are required to represent the input using a fixed-length code? Of course, you want to use as few bits as possible. **Sol.** 3 * 27 =

5. (15 points) Interval Selection Problem (ISP). In the ISP, we are given n intervals $(s_1, f_1), (s_2, f_2), ..., (s_n, f_n)$, and are asked to find a largest subset of intervals that are mutually disjoint. For simplicity, let's assume that all s, f values are distinct. We refer to s_i and f_i as interval i's start and finish times, respectively. Also assume that intervals are ordered in increasing order of their finish times. Prove that there exists an optimal solution that includes the first interval (s_1, f_1). **Sol.** See the lecture slides or textbook.

6. (15 points) The following is a pseudocode of a DP based algorithm for the rod cutting problem. So, at the end, the code has successfully computed $r[0...n]$. Here, for simplicity, we assume that the price array $p[1...n]$ can be accessed globally. Modify the pseudocode so that we print out an optimal cutting at the end. In other words, output the length of each piece in an optimal cutting of a rod of length n (that gives the maximum revenue).

```
Bottom-Up-Cut-Rod(n)
1. Let r[0..n] be a new array
2. r[0] = 0
3. for j = 1 to n
4.    q = -infinity
5.   for i = 1 to j
6.        q = max(q, p[i] + r[j-i])
7.   r[j] = q
```

Sol. See the lecture slides or textbook.

7. (15 points) The following is a pseudocode of a bottom-up DP algorithm for computing nth Fibonacci number. Give a pseudocode of a *top-down* DP algorithm for the problem, which uses memoization.

```
int F(n)
  Array A[0 ... n]
  A[0] = 0, A[1] = 1
  for i = 2; i <= n ; i++
     A[i] = A[i-1] + A[i-2]
  return A[i]
```

Sol. See the lecture slides or textbook.

8. (40 points) (Note: This is a bonus problem. This problem is for students who challenged themselves hard, so they could solve new problems they have never seen before. This problem is not for collecting partial points. So, unless your answer is almost correct, you will likely get 0 points.)

   The board cutting problem is a 2D generalization of the rod cutting problem. In the board cutting problem, we are given a m-by-n flat board – in other words, the board has width m and height n. Also, we are given a price table $p[i, j]$, $1 \leq i \leq m$ and $1 \leq j \leq n$, where $p[i, j]$ is the market price of a i-by-j board. We are only allowed to split a board horizontally or vertically. The width and height of the resulting boards must be integers. Here, our goal is to compute the maximum revenue we can get out of the given m-by-n board by iteratively cutting the board. (30 points) Give a DP for this problem. Of course, your DP must run in polynomial time in m and n. Your recursion and DP table must be clearly presented. Also state in which order you will fill out the DP table and which value you need to return at the end as the answer. (10 points) Analyze the running time of your DP.

   Sol. Let $r[i, j]$ denote the max revenue from a i by j board. Set $r[i, j] = 0$ if $i = 0$ or $j = 0$, which forms the base cases.

   Otherwise,

$$r[i, j] = \max \left\{ \max_{1 \leq k \leq i} (p[k, j] + r[i - k, j]), \max_{1 \leq k \leq j} (p[i, k] + r[i, j - k]) \right\}.$$

   We can fill out the table in row major order, and return $r[m, n]$ as the optimal revenue.

   Now let's discuss the RT analysis. There are mn entries to fill out. To fill out each entry, particularly $c[i, j]$, we need $O(\max\{i, j\})$ time, which is at most $O(\max\{m, n\})$. Therefore, the DP RT is $O(mn \max\{m, n\}))$.