Algorithm

2^{h-1} n

$2^{k} = 2n$

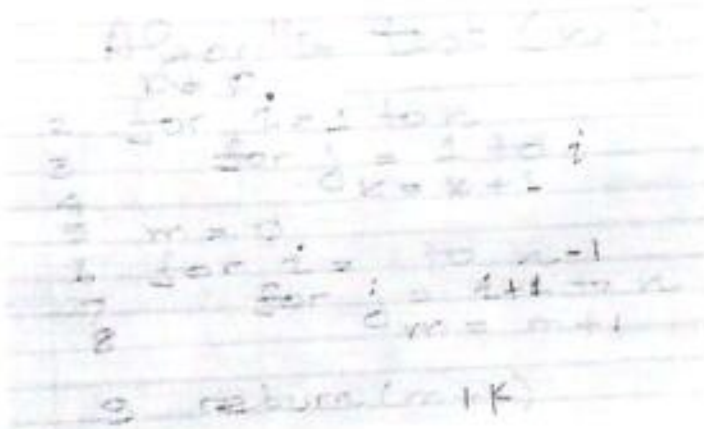
Need to solve

1. Previous Comprehensive Questions
   Fall **19, 18**, 17, *16, 15*
   Spring **19, 18,** 17, *16, 15*

2. Dr. Zheng's Course Questions
   Fall 19, 18

3. Dr. Jin's Course Questions
   Spring 19, Spring 18

1.	Find tight bound for $\sum_{i=1}^{i=n} i^k$ where k is a constant.	$n.n^k = \theta(n^{k+1})$ Need a verified solution
2.	Find tight bound for $\sum_{i=1}^{i=n}$ i Log(i)	$\theta(n^2Lg(n))$
3.	Divide & Conquer method - Conditions - Concept - When can be applied to solve problem - Can it be applied for a) MST b) 0-1 knapsack c) Merge Sort? Why? Or Why not?	Fall 14
4.	What is the lower bound for comparison based sorting algorithms?	
5.	What is the time complexity of insertion sort algorithm?	Fall 15 (3)
6. slide	Define - Upper and tight bound - Time and space complexity	
?? 7.	In which way the average time bound add more value to the tight bound?	
8.	Briefly describe quick sort algorithm. Mention the best and worst case time complexity of it.	Fall 16
9.	Show that for real constants a and b, b>0 $(n+a)^b = \theta(n^b)$ picture	Solution Fall 15

<table>
<tr><td>10.</td><td>Time complexity of algorithm (i) is $O(n^2)$ and algorithm (ii) is $O(n^3)$.
- Which you choose?
- For algorithm (i) → T9n) = $1000n^2$ and for (ii) $T(n) = 2n^3$.
Which you choose and why?</td><td>Fall 14
Spring 14</td></tr>
<tr><td>11.
note</td><td>(a) To what extent the asymptotic upper bound and lower bound provide insight on running time of an algorithm.
(b) Compare and contrast asymptotic tight bound to average running time of an algorithm.
(c) Consider the pseudo code of an algorithm given below
c.1 What does the value K in Line 4 denote?
c.2 What does the value m in Line 8 denote?
c.3 When the algorithm terminates, what does the value of m+K in Line 9 denote?
c.4 Find the asymptotic tight bound of Algorithm Test below.
</td><td>Spring 2019
Long Q3

Log n</td></tr>
<tr><td>12.
note</td><td>2. In terms of run time efficiency, compare and contrast quick sort and merge sort? What is the best and the worst case time complexity of quick sort algorithm. Also state under what condition one may expect these two extreme cases?</td><td></td></tr>
</table>

```
k = 0
for i = 1 to n
    for j = 1 to i
        k = k + 1

m = 0
for i = 1 to n - 1
    for j = i + 1 to n
        m = m + 1

return (m + k)
```

Find Big-O/Theta

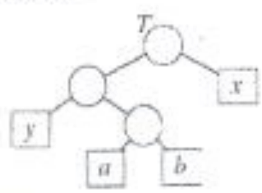
13	$T(n) = 3n^3 + 20n^2+1000$	
14	$T(n) = 2T(n/8) + n^{1/3}$	
15	$T(n) = Log(n!)$	
16	$T(n) = T(n-2) + 2Log_2(n)$	Without master method
17	$T(n) = 2T(n^{1/2}) + Log_2(n)$	Solution Fall16 (3)
18	$T(n) = T(n/3) + T(2n/3) + cn$	Both Big-O and Theta Using recurrence tree
19	$T(n) = nLog(n!) + 3n^2 + 2n - 10000$	Solution: $O(n^2 logn)$
20	$T(n) = T(n/4) + T(3n/4) + cn$	
21	$T(n) = T(n/2)+n$	Without master method

Part-2 Data Structures

Simulations ***

1.	B-Tree	Fall 19, 18, 16, 15 Spring 19, 18
2.	Fibonacci Mean Heap	Spring 18
3.	Double Hashing	

Theoretical/Conceptual

4.	Proof: For any n-key B-tree -→ Log$_t$ (n+1/2)	Fall 19 (4), 15 (8)
5.	Proof: Open address hashing load factor = n/m Given that for an open-address hash table with load factor $\alpha = n/m < 1$, the expected number of probes in <u>unsuccessful search</u> under uniform hashing is at most $1/(1-\alpha)$, how do you prove the expected number of probes in a <u>successful probe</u> under uniform hashing being at most $(1/\alpha) \cdot \ln(1-\alpha)^{-1}$? (Just give a proof sketch, explaining how many probes are needed to locate existing keys.)	Spring 18 (2)
6.	Proof: Prefix Code → Lemma Sketch a proof of the Lemma below, using the tree provided. Let C be an alphabet in which each character $c \in C$ has frequency $c.freq$. Let x and y be two characters in C having the lowest frequencies. Then there exists an optimal prefix code for C in which the codewords for x and y have the same length and differ only in the last bit.  Part. 3	Spring 18 (3)
7.	Hashing: Multiplication method description + figure	Fall --, 15
8.	Define height balance binary tree. Pseduocode to determine if a tree is height balanced. Tight bound for your algorithm	Spring 18, 17 Solution on Spring18 (5)

9.	How perfect hashing works? Steps in inserting with the key in Cuckoo Hashing What is the situation when a new key can not be inserted in cuckoo hasing.?	Fall 16 (9)
10.	Property of min and max heap. What is the preferred data structure of implementing binary heap? Time complexity of merging two different min heaps with size n and m. [O(n+m)]	Spring 18, 17 Solution at Spring 18 reverse pages
11.	Suppose you have [illegible] N objects stored in an array in the ascending order of key values. [illegible] a. Describe an efficient algorithm with the pseudocode that helps you to search for the object given the key (the operation must return a null value if the key is not in the array). b. Obtain the big O expression for your algorithm.	[illegible]
12.	Given two hash functions of h_1 and h_2 for Cuckoo hashing under two tables, T_1 and T_2: a) Describe the steps involved in inserting a record with the key of [illegible] b) Cuckoo hashing can be analyzed by the Cuckoo graph, whose nodes denote table entries and links connect pairs of nodes where given keys can be held. State when a new key can be inserted successfully based on the Cuckoo graph.	
13.	The utilization efficiency of a hash table depends heavily on its hashing function(s) employed. Describe with a diagram to illustrate how a multiplication method of hashing works on a machine with the word size of w bits for a hash table with 2^p entries, $p < w$. Explain briefly how Cuckoo hashing works under two hash functions of h_1 and h_2.	***

Part-3 Dynamic and Greedy Programming

LCS → MiO
Jonson → "

Simulations ***

1.		
2.	Constructing Huffman Codes ***	Fall 19, 18 Spring 19, 16
3.	Matrix Chain Multiplication	Fall 19, Fall 18
4.	OBST + Expected Cost	Fall 18, 16, 14 Spring 17, 14

Theoretical/Conceptual

5.	Briefly define greedy strategy? /What is the strategy behind greedy algorithm? Will it always yield optimal? Why if yes or why not if no?	Fall 18, 16, 14
6.	CUT-ROD: T[0]1, Solve T[n]=1+sum<j=0>(T[j])	
7.	Describe Dynamic Programming. What are the conditions for DP? Can it be applied to solve any optimization problem? Explain why or why not.	Fall 18
8.	How DP is used to construct an OBST?	Fall --, 14 Spring 14
9.	Knapsack Code Filling (Dr. Miao Jin) with table filling *** if (mem[n][w]) → subset sum, → minimum-coin → knapsack mem[n][w] = result.	

<table>
<tr>
<td>10</td>
<td>

LCS and MCM Conceptual Question

(Problem is related to dynamic programing formulation and optimal sub structure)

a. Explain what do you understand by "principle of optimality."

b. Consider the problem of finding the longest common subsequences (LCS) in a pair of sequences, namely, $X(x_1, x_2, \ldots, x_m)$ and $Y(y_1, y_2, \ldots, y_n)$.

b1. A brute force approach is to generate all possible subsequences of X and see whether it is also a subsequence of Y. What is the number of possible subsequence of X?

b2. Obtain the optimal substructure of an LCS.

c. Consider a chain matrix multiplication problem, $m_1 \times m_2 \times m_3 \times \ldots \times m_n$. It is associative, but not commutative.

c1. A brute force approach is to generate all possible ways of parenthesizing the matrix chain and compute the total number of operations. What is the number of possible grouping of the matrix chain?

c2. Obtain the structure of the optimal parenthesization and then the recursive definition of the minimum cost of parenthesizing the product $m_i, m_{i+1}, \ldots, m_j$.

</td>
<td>Spring 17 Long Q1</td>
</tr>
<tr>
<td>11</td>
<td>

Explain Principle of Optimality.

Describe basic rules that satisfy Principle of Optimality and domain related constraints-

1. 0-1 Knapsack problem
2. Pairwise Shortest Path problem
3. Chain Matrix multiplication problem

</td>
<td>Spring 15 Long 2</td>
</tr>
<tr>
<td>12</td>
<td>

The divide and conquer strategy (D&C) has been used to solve problem effectively to reduce the overall computational cost to certain types of problems.

(a) Which conditions have to be satisfied for D&C to solve such problems successfully? (Clearly state)

(b) Suppose the size of a problem involved in D&C is $n = 2^k$. Let the cost in dividing the problem into an equal size is constant and the time to combine solutions to sub-problems is linear. Write the recurrence relations and then find the tight bound in solving such problems using D&C.

</td>
<td>Mirza bhai</td>
</tr>
</table>

Part-4 Graph

$BFS = O(V+E)$
$DFS = O(2V+E)$

Simulations ***

1. BFS + DFS + Topological Sorting
2. Prim's
3. Krushkal
4. Dijkstra $O((V+E)\log V)$, $O(V^2+E)$
5. Floyd Warshal $O(n^3)$
6. Bellman Ford $O(VE)$
7. Ford Fulkerson + Min Cut
8. Edmonton Karp $O(VE)$
9. Johnson's

Theoretical/Conceptual

10. Time Complexity of Dijkstra if (i) candidate vertexes are kept in a binary min-heap (ii) are kept in an array
11. Define/Describe MST.
12. Salient features of MST.
13. Name two algorithms for obtaining MST. Which one do you prefer and why?
14. Sketch an algorithm to obtain MST
15. Prim's v/s Krushkal
16. Define Strongly Connected Components (SCC) as applied to directed graphs.
17. Describe some potential applications of SCC.
18. SCC graph cyclic or acyclic? Justify your answer.
19. Give pseducode for obtaining SCC.
20. Given a weighted directed graph $G = (V, E, w)$ and a shortest path P from s to t, if the weight of every edge is doubled to produce $G^* = (V, E, w^*)$, is P still a shortest path in G^*? Explain your reasoning behind your answer.

Part-5 Other Topics

1.	Define/Compare and Contrast P, NP, NP-Complete and NP-Hard. Draw venn diagram.	****
2.	Suppose, a single NP-Complete problem is solved in polynomial algorithm, what can you state about the entire NP-Complete class as well as the NP-Hard class?	
3.	3P-SAT Problem related	Spring 2017 Long Q2 Least common
4.	2-proof component of NP-Complete 2-approximation of TSP	
5.	Conceptual Question	Fall 19 (7) Spring 19 (6)
6.	Show 0-1 Knapsack problem belongs to NP-Class	9/11

unknown

7.

(a) Define the following classes of a decision problem: P, NP, and NP-completeness.

(b) Consider the 0-1 knapsack problem with n objects each with its respective pre-defined profit. The objective is to maximize the total profit that can be accommodated into a container of capacity W. Defining appropriate notations for weight and profit of objects, formulate the problem.

(c) Convert of the problem that you have defined in (b), into a decision problem.

(d) Show the problem that you have defined in (c) belongs to NP-class.

(e) Does the problem in (d) belong to the P-class or NP-completeness. (Justify your answer.)

(f) If principle of optimality be applicable to solve the problem defined in (c), formulate it. Otherwise, explain why not.

(g) What would be your explanation, if 0-1 knapsack problem is solved by dynamic programming in polynomial time?

a. Compare and contrast P, NP, NP-complete and NP-hard.

b. Based on current conjecture, draw a venn diagram to show the relationship among these classes of problem.

c. Suppose there are n clauses and m propositions in a given 3p-sat problem. How many possible interpretations are there? What is the time complexity of testing the satisfiability of a given interpretation? What is the time and space complexity of testing the satisfiability of the clauses.

d. 3-p sat problem is NP-complete, but still people have published papers by applying heuristics strategy and showing that they were able to solve it with large number of distinct propositions (say 100) and large number of clauses (say 200). To avoid any bias, they have generated the clauses and the set of propositions randomly. How would you start investigating their results? Can their results be generalized?

e. Suppose a single NP-complete problem is solved in polynomial algorithm, what can you state about the entire NP-complete class as well as the NP-hard class.

8.

a) Explain what do you understand by "principle of optimality" in the context of dynamic programing.

b) Characterize 0-1 knapsack problem in terms of objective function, constraints and the time and space complexity. (Assume that there are n objects. Suppose an object i has weight w_i and profit p_i. The overall capacity of the container is W).

c) Show the 0-1 knapsack problem belong to NP-class.

d) Does it belongs to P-class? (provide an explanation accordingly)

e) Write down the basic rule that satisfy the principle of optimality and domain related constraints to the following problems:
   e.1) 0-1 knapsack problem.
   e.2) Pairwise shortest path problem.

9.

a) Briefly describe NP-class, P-class, NP-complete and NP-hard.

b) Show the conjectured relationship among the classes NP-class, P-class, NP-complete and NP-hard.

c) Show that sorting n objects with integer key values belongs to NP-class.

d) Provide the steps involved in showing whether a problem belongs to NP-complete or not.

e) Illustrate the steps in step d by showing 3 proposition satisfiability (3-p sat) problem belongs to NP-complete.

f) Provide a pseudo code that attempt to solve 3-p sat problem heuristically.

10.