

**CS-2009: Design & Analysis
of Algorithms(SOLUTION)**

Serial No:
Final Exam
Total Time: 3 Ho
Total Marks: 100

Saturday, 06th January, 2024

Course Instructors

Dr. Ramoza Ahsan, Ms. Laralb Afzaal

Signature of Invigilator

Student Name

Roll No.

Course Section

Student Signature

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Attempt on question paper. Read the question carefully, understand the question, and then attempt it.
2. No additional sheet will be provided for rough work.
3. Verify that you have twenty four(24) different printed pages including this title page. There are six (6) questions.
4. Calculator sharing is strictly prohibited.
5. Use permanent ink pens only. Any part done using soft pencil will not be marked and cannot be claimed for rechecking.
6. Ensure that you do not have any electronic gadget (like mobile phone, smart watch, etc.) with you.

	Q-1	Q-2	Q-3	Q-4	Q-5	Q-6	Total
Marks Obtained							
Total Marks	20	35	15	10	10	10	100

National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

Question 1 MCQs [20 Marks]

Please write the correct option in the table given at the start of the MCQs. No marks will be awarded if table is not filled. Overwriting or multiple answers will result in zero marks.

MCQ Number	Correct Answer
1	C
2	C
3	A
4	A
5	B
6	B
7	C
8	A
9	D
10	C
11	A
12	D
13	D
14	C
15	B
16	A
17	A
18	D
19	D
20	B

MCQ Number	Correct Answer
21	A
22	B
23	C
24	A
25	B
26	D
27	A
28	D
29	C
30	D
31	C
32	C
33	D
34	B
35	A
36	D
37	D
38	D
39	A
40	A

Q1: What will be the time complexity of the following code?

```
if(i<j){
    if(j==0){
        j++
    }
    else
        j--
}
```

- a) $O(n^2)$
- b) $O(n)$
- c) $O(1)$
- d) $O(\log n)$

Q2: What will be the time complexity of the following code?

```
for(i=n/4;i<=n;i++){
    for(j=1;j<=n;j=j*4){
        a++
    }
}
```

- a) $O(n \lg n)$
- b) $O(n^2)$
- c) $O(n \log_4 n)$
- d) $O(n/4 * n)$

Q3: What is the worst case complexity of bubble sort?

- a) $O(n^2)$
- b) $O(n)$
- c) $O(\lg n)$
- d) $O(n \lg n)$

Q4: Quick sort follows Divide and Conquer strategy

- a) True
- b) False

Q5: Merge sort can be implemented using $O(1)$ auxiliary space

- a) True
- b) False

Q6: Choose the incorrect statement about merge sort from the following

- a) It is a comparison based sort
- b) It is not an in place algorithm
- c) It is a stable algorithm
- d) None of the above

Q7: Which of the following sorting techniques is most efficient if the range of input data is not significantly greater than a number of elements to be sorted?

- a) Selection sort
- b) Bubble sort
- c) Counting sort
- d) Insertion sort

Q8: What is the average time complexity of counting sort where k is the range of input numbers?

- a) $O(n+k)$
- b) $O(n^2)$
- c) $O(nk)$
- d) $O(k)$

Q9: Which of the following is/are property/properties of a dynamic programming problem?

- a) Optimal substructure
- b) Overlapping subproblems
- c) Greedy approach
- d) Both optimal substructure and overlapping subproblems

National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

Q10: If a problem can be solved by combining optimal solutions to non-overlapping problems, the strategy is called

- a) Dynamic Programming
- b) Greedy
- c) Divide and Conquer
- d) None of the above

Q11: A greedy algorithm cannot always find an optimal solution of which of the following problem?

- a) 0/1 Knapsack problem
- b) Data compression using Huffman Coding
- c) Fractional Knapsack problem
- d) None of the above

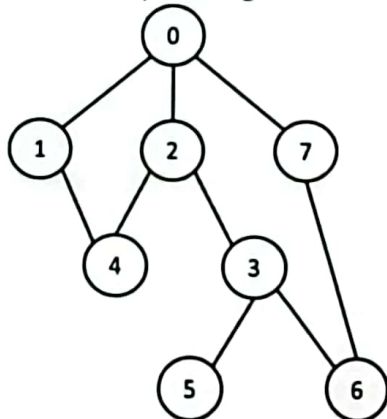
Q12: Consider the two matrices P and Q which are 10 x 20 and 20 x 30 matrices respectively. What is the number of scalar multiplications required to multiply the two matrices?

- a) 10*20
- b) 20*30
- c) 10*30
- d) 10*20*30

Q13: Consider the brute force implementation in which we find all the possible ways of multiplying the given set of n matrices. What is the time complexity of this implementation?

- a) $O(n!)$
- b) $O(n^3)$
- c) $O(n^2)$
- d) Exponential

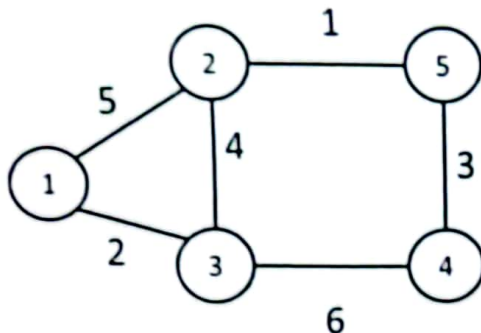
Q14: What will be the order of traversal of nodes in the following graph if we run BFS (breath first search) starting from node 0 on the graph (BFS will pick smaller neighbors first)?



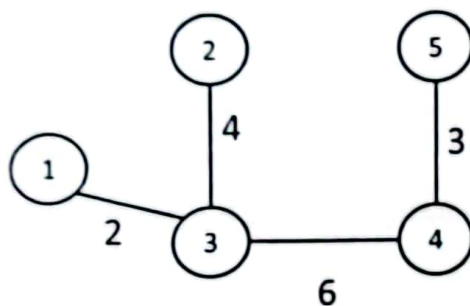
- a) 01234567
- b) 01274365
- c) 01274356

d) 01423567

Q15: Given the following graph:



Is the following tree a valid Minimum spanning tree of it?



- a) Yes
- b) No

Q16: Complexity of Prim's algorithm (using the binary heap) for finding minimum spanning tree is?

- a) $O(E \lg V)$
- b) $O(E^2)$
- c) $O(VE)$
- d) $O(V)$

Q17: What is the recurrence equation for Merge Sort?

- a) $T(n) = 2T(n/2) + O(n)$
- b) $T(n) = T(n/2) + O(n)$
- c) $T(n) = T(n/2) + c$
- d) $T(n) = O(n)$

Q18: Which of the following is an example of linear time sorting algorithm?

- a) Radix sort
- b) Count Sort
- c) Bucket Sort

d) All of the above

Q19: What is the complexity of Build heap?

- a) $O(n \log n)$
- b) $O(n^3)$
- c) $O(n^2)$
- d) $O(n)$

Q20: What is the complexity of extracting maximum value from max-heap?

- a) $O(n \log n)$
- b) $O(\log n)$
- c) $O(n)$
- d) $O(n^2)$

Q21: Counting sort performs number of comparisons between input elements.

- a) 0
- b) n
- c) $n \log n$
- d) n^2

Q22: You are given infinite coins of denominations $v_1, v_2, v_3, \dots, v_n$ and a sum S . The coin change problem is to find the minimum number of coins required to get the sum S . This problem can be solved using

- a) greedy algorithm
- b) dynamic programming
- c) divide and conquer
- d) backtracking

Q23: For merging two sorted lists of size m and n into sorted list of size $m+n$, we require comparisons of

- a) $O(m)$
- b) $O(n)$
- c) $O(m+n)$
- d) $O(\log m + \log n)$

Q24: How many passes are required to sort a file of size n by bubble sort method?

- a) $N-1$
- b) $N/2$
- c) N
- d) None of the above

Q25: What is the time complexity of the following code snippet in C++?

```
void solve() {  
    string s = "scaler";  
    int n = s.size();  
    for(int i = 0; i < n; i++) {  
        s = s + s[i];  
    }  
    cout << s << endl;  
}
```

- a) $O(n)$
- b) $O(1)$
- c) $O(n^2)$
- d) $O(\log n)$

Q26: Which of the following is false?

- a) the spanning trees do not have any cycles
- b) MST have $n - 1$ edges if the graph has n edges
- c) edge e belonging to a cut of the graph if has the weight smaller than any other edge in the same cut, then the edge e is present in all the MSTs of the graph
- d) removing one edge from the spanning tree will not make the graph disconnected

Q27: Kruskal's algorithm is best suited for the sparse graphs than the prim's algorithm.

- a) true
- b) false

Q28: Which of the following is the most commonly used data structure for implementing Dijkstra's Algorithm?

- a) max priority queue

- b) stack
- c) circular queue
- d) min priority queue

Q29: Fractional Knapsack problem is solved most efficiently by which of the following algorithm?

- a) Divide and conquer
- b) Dynamic programming
- c) Greedy algorithm
- d) Backtracking

Q30: Which of the following is the recurrence relation that describes processing time $T(n)$ of the following recursive method?

function finalExam (n)

if (n > 1)

print 'A'

finalExam(n / 3)

for i = 1 to n

print 'B'

end for

finalExam (n / 3)

- a) $T(n)=T(n/3) + T(n/3)$
- b) $T(n)=2T(n/3) + O(1)$
- c) $T(n)=T(1) + T(n/3) + O(n) + T(n/3)$
- d) $T(n)=2T(n/3) + O(n)$

Q31: Which of the following statements is/are true?

1. Adjacency list representation is better for sparse graph than adjacency matrix representation.

National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

2. Finding whether there is an edge between any two nodes in a graph is easier in Adjacency list representation.

3. Adding a vertex in adjacency list representation is easier than adjacency Matrix representation.

- a) I, II and III
- b) II and III
- c) I and III
- d) I only

Q32: Dynamic Programming is more beneficial than Divide and Conquer when?

- a) The problem is easily divided into subproblems
- b) The problem has a recursive structure
- c) There are overlapping subproblems
- d) The problem has optimal substructure

Q33: In Quick Sort, the worst-case time complexity occurs when?

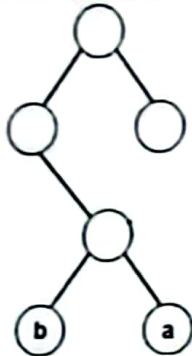
- a) The pivot is chosen as the median element
- b) The pivot is chosen as the first element
- c) The pivot is chosen as the last element
- d) The pivot is chosen poorly, leading to skewed partitions

Q34: Which of the following algorithms is the best approach for solving Huffman codes?

- a) Exhaustive search
- b) Greedy algorithm
- c) Brute force algorithm

d) Divide and conquer algorithm

Q35: From the following given tree, what is the code word for the character 'a' if Huffman codes are used?



a) 011

b) 010

c) 100

d) 101

Q36: What will be the recurrence equation for quicksort if split is done in the 9:1 ratio, i.e. 9 elements in the left partition and 1 element in the right partition?

a) $T(n) = T(n-9) + T(n-1) + O(n)$

b) $T(n) = T(9n/10) + T(n/10) + O(n^2)$

c) $T(n) = T(9n/10) + T(n/10) + O(1)$

d) $T(n) = T(9n/10) + T(n/10) + O(n)$

Q37: What is the time complexity of the divide and conquer algorithm used to find the maximum sub-array sum?

a) $O(n)$

b) $O(\log n)$

c) $O(n^2)$

d) $O(n \log n)$

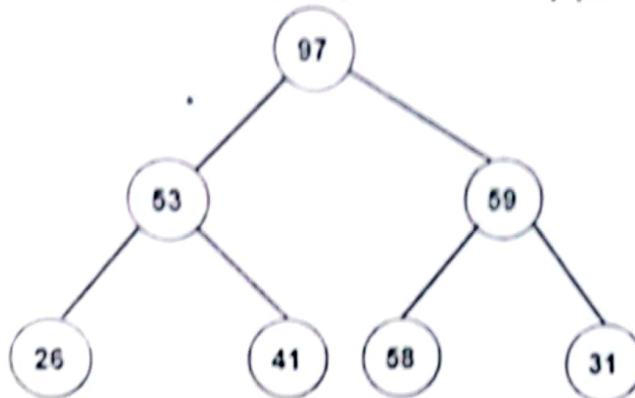
National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

Q38: Consider the following heap after buildheap phase. What will be its corresponding array?



- a) 26,53,41,97,58,59,31
- b) 26,31,41,53,58,59,97
- c) 26,41,53,97,31,58,59
- d) 97,53,59,26,41,58,31

Q39: Kruskal's algorithm is used to _____?

- a) find minimum spanning tree
- b) find single source shortest path
- c) find all pair shortest path algorithm
- d) traverse the graph

Q40: Which one is a smiley?

- a) 😊
- b) ☹️
- c) None of the above

National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

Question 2a: For each sorting algorithm below, give the asymptotic worst-case tight bound using "Big-O" notation. You should give the tightest and simplest bound possible. You do NOT need to explain your answer. [4 points]

Merge Sort	Insertion Sort	Selection Sort	Heap Sort
$O(n \log n)$	$O(n^2)$	$O(n^2)$	$O(n \log n)$

Question 2b: In a competition, four different functions are observed. All the functions use a single for loop and within the for loop, the same set of statements are executed. Consider the following for loops:

A) for($i = 0$; $i < n$; $i++$)

B) for($i = 0$; $i < n$; $i += 2$)

C) for($i = 1$; $i < n$; $i *= 2$)

D) for($i = n$; $i > -1$; $i /= 2$)

If n is the size of input(positive), which function is the most efficient (if the task to be performed is not an issue)? [2 points]

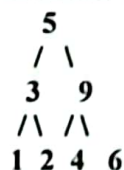
C is correct. Time complexity is $O(\log n)$

The time complexity of second for loop is $\Theta(n/2)$, equivalent to $\Theta(n)$ in asymptotic analysis.

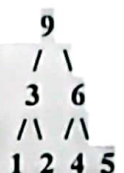
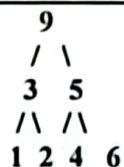
The time complexity of third for loop is $\Theta(\log n)$.

The fourth for loop doesn't terminate.

Question 2c: Consider the following binary tree: [2 points]



What will the tree look like after a call to Heapify on the root node?



Question 2d: Consider the following recursive algorithm.


```
def Algo(n){
  if n <= 1
    return n;
  else
    return Algo(n - 1) + Algo(n / 2);
}
```

Write the recurrence for the algorithm [3 points]

$$T(n) = T(n-1) + T(n/2) + O(1)$$

Question 2e: Solve above recurrence relation using the iteration method [5 points].

To solve this recurrence relation using the iteration method, let's expand it step by step:

$$\begin{aligned} T(n) &= T(n-1) + T(n/2) + O(1) \\ T(n-1) &= T(n-2) + T((n-1)/2) + O(1) \\ T(n/2) &= T(n/2-1) + T(n/4) + O(1) \end{aligned}$$

Continuing this pattern, we can generalize the relation as:

$$T(n) = T(n-k) + T(n/2^k) + k \cdot O(1)$$

Let's find the base case when $n=1$:

$$T(1) = T(n-k) + T(n/2^k) + k \cdot O(1)$$

When $n=1$, it implies $k = \log_2 n$, and $T(1)$ represents the base case which is 1.

So, when $n=1$, $T(1) = 1$.

Now, let's increment k until it reaches n :

$$\begin{aligned} T(n) &= T(n-k) + T(n/2^k) + k \cdot O(1) \\ T(n) &= T(1) + T(1) + \log_2 n \cdot O(1) \\ T(n) &= 1 + 1 + \log_2 n \cdot O(1) \\ T(n) &= 2 + \log_2 n \cdot O(1) \end{aligned}$$

Time complexity is $O(\log n)$

National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

Question 2f: [5 points]

Suppose that you are required to choose one of the following 2 algorithms:

1. Algorithm A solves the problem of size n by dividing it into 5 sub-problems of size $n/2$, recursively solves each sub-problem, and then combines the solutions in linear time.
2. Algorithm B solves the problem of size n by dividing it into nine sub-problems of size $n/3$, recursively solves each sub-problem, and then combines the solutions in $O(n^2)$ time.

What is the asymptotic time complexity of each algorithm? Which one would you choose?

Considering the asymptotic time complexities, Algorithm B (with a time complexity of $O(n^2)$) would generally be preferable for large problem sizes since it has a lower exponent in its polynomial time complexity compared to Algorithm A. Therefore, for larger problem instances, Algorithm B might perform better in terms of time complexity.

Solve following using Masters Theorem

Question 2g: [3 points]

$$T(n) = 64T(n/8) - n^2 \cdot \log n$$

Does not apply as $F(n)$ is not positive

Question 2h: [3 points]

$$T(n) = 3T(n/2) + n$$

$\Theta(n \lg 3)$, case 1

Question 2i: [3 points]

$$T(n) = 4T(n/2) + n^2$$

$\Theta(n^2 \log n)$, case 2

Question 2j: [3 points]

$$T(n) = 0.5T(n/2) + n^4$$

$$a=0.5 \quad b=2 \quad h=2$$

$$f(n)=n^4$$

$\Theta(n^4)$, case 3

Question 2k: [2 points]

Given an array of n integers, each belonging to $\{-1, 0, 1\}$, how we can sort the array in $O(n)$ time in the worst case. (Just write the name of the sorting technique that can be used and how you would modify it. You don't need to write the Pseudocode)

Use counting sort, add 1 to every number, sort the numbers using counting sort and subtract 1 from every number to get original numbers.

National University of Computer and Emerging Sciences

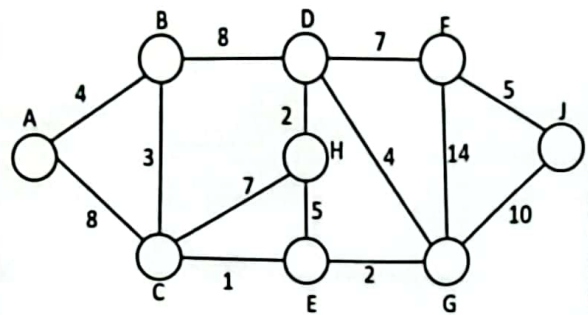
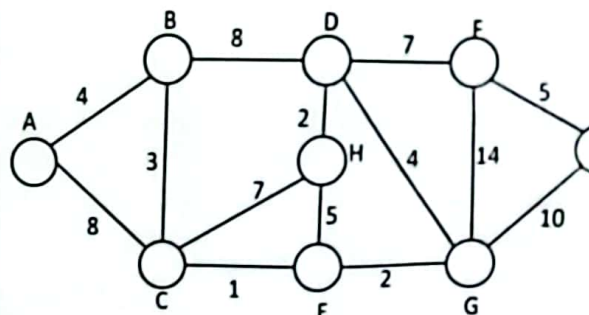
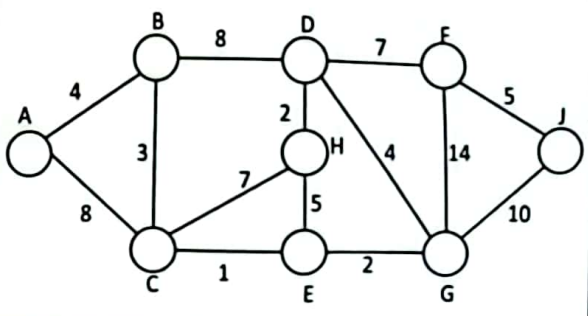
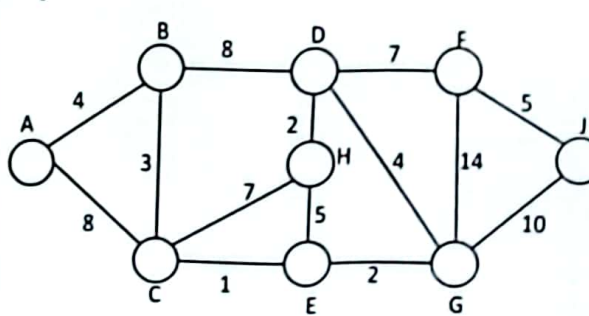
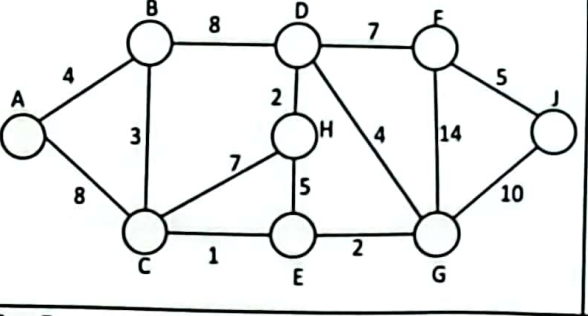
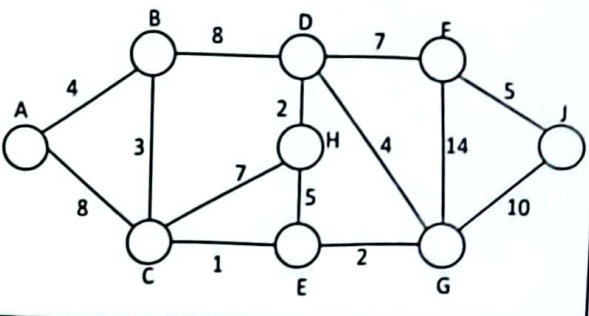
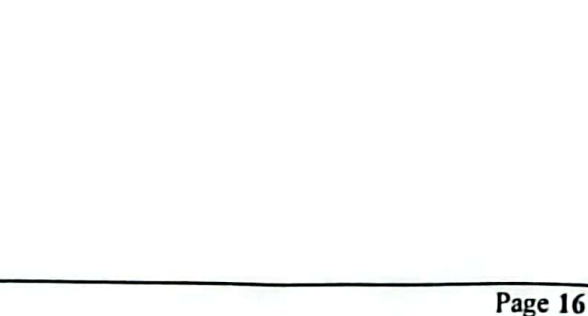
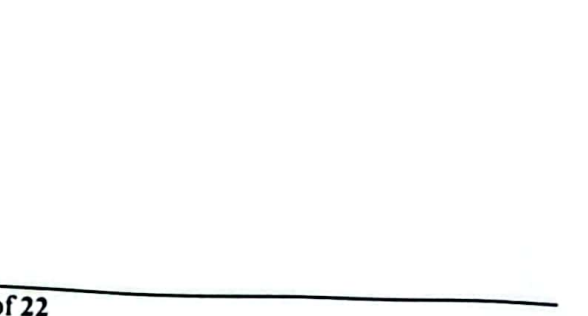
FAST School of Computing

Fall-2023

Islamabad Campus

Question 3 Shortest Paths [15 Marks]

Question 3a: Find the shortest path in the following graph from A to all the nodes using Dijkstra algorithm. Write the distance of the node inside the node in each step (starting with 0 in node A). Once a node is done add a tick over it. Show your working of each step. [8 points]

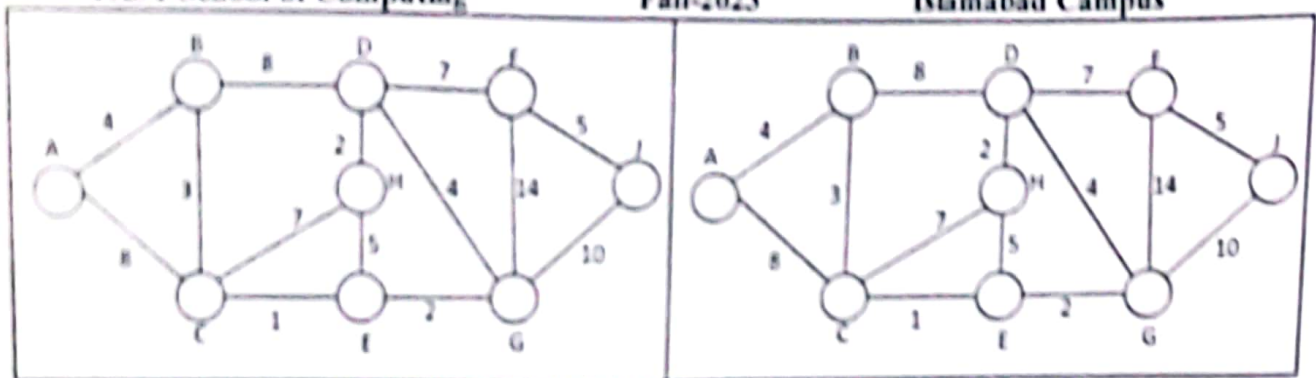
<p>Step1</p> 	<p>Step2</p> 
<p>Step3</p> 	<p>Step4</p> 
<p>Step5</p> 	<p>Step6</p> 
<p>Step7</p> 	<p>Step8</p> 

National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus



Solution:

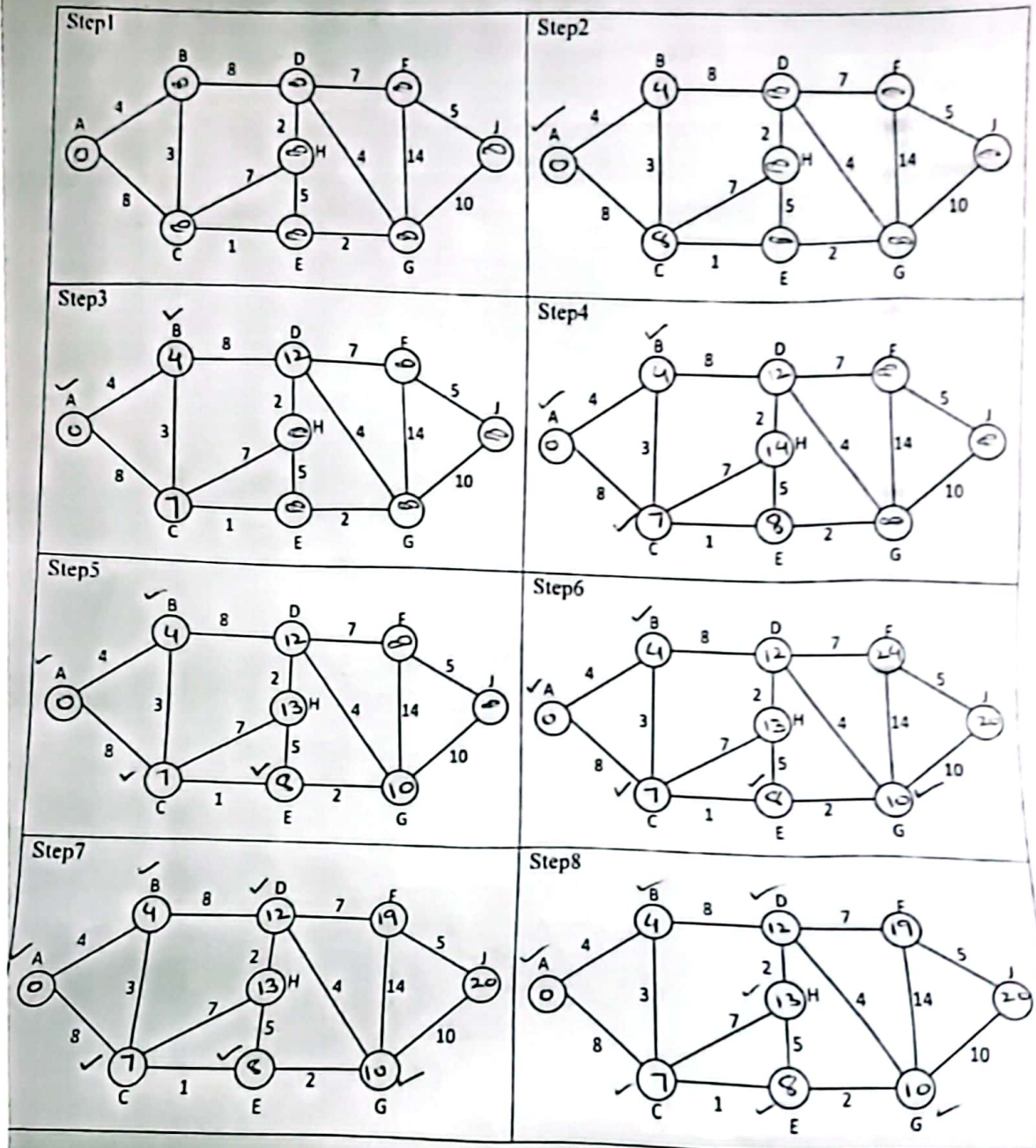
National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

Once a node is done add a tick over it. Show your working of each step. [8 points]



Page 16 of 24

No change afterword

Question 3b: What is the shortest path from A to H in the above graph (Write the name of nodes in the shortest path from A to H) [2 points]

ABCEH

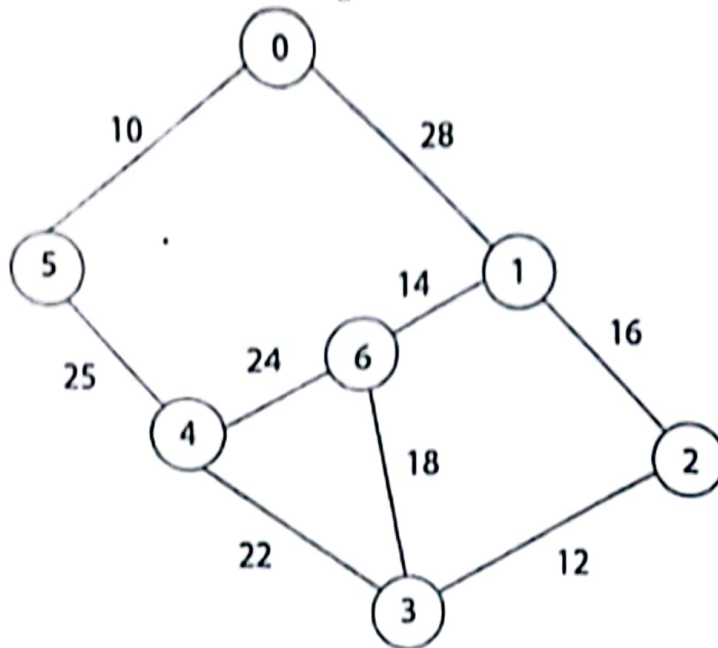
National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

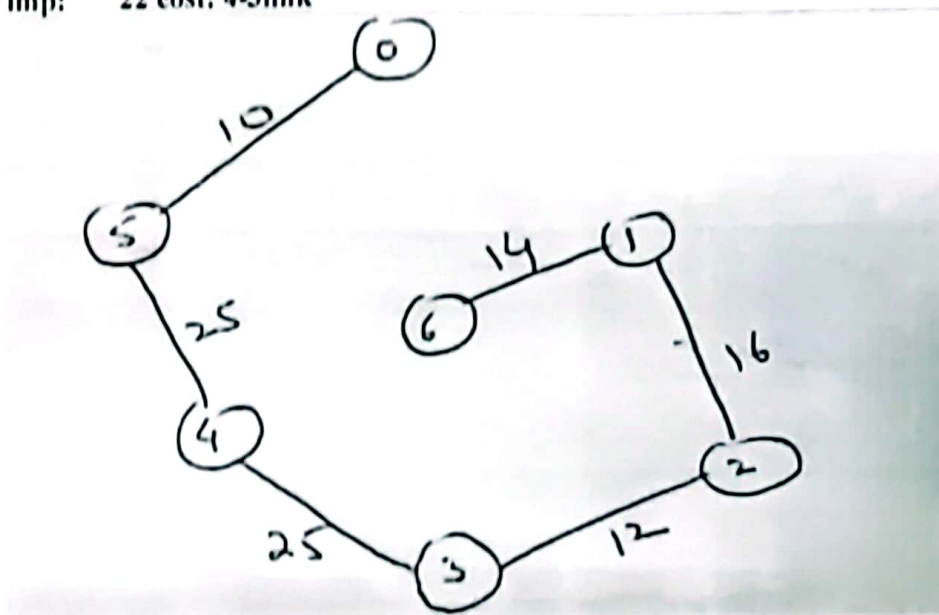
Islamabad Campus

Question 3c: Find the minimum spanning tree (MST) using Prim's algorithm for the following graph where 0 is the starting root node. (Draw the MST and the cost of the MST) [5 points]



MST: [4 points]

Imp: 22 cost: 4-3link



What is the Cost of the MST: [1 point] 99

National University of Computer and Emerging Sciences

FAST School of Computing

Fall-2023

Islamabad Campus

Question 4 Dynamic Programming [10 Marks]

For the given set of items and a knapsack capacity of 5 kg, find the optimal solution for the 0/1 knapsack problem using a dynamic programming approach.

Item	Weight	Value
1	2	3
2	3	4
3	4	5
4	5	6

Question 4a: Write the recursive solution equation for solving 0/1 knapsack problem using dynamic programming [1 point]

$$V[i, j] = \max(V[i-1, j], v_i + V[i-1, j - w_i])$$

Question 4b: Fill the V matrix below (Show your working to get full credit) [6 points]

V[i, w]	W = 0	1	2	3	4	5
i = 0	0	0	0	0	0	0
i = 1	0	0	3	3	3	3
i = 2	0	0	3	4	4	7
i = 3	0	0	3	4	5	7
i = 4	0	0	3	4	5	7

Question 4c: What is the optimal value of the knapsack? [1 point]

7

Question 4d: Which items should be picked based on the optimal value? [2 point]

Item 1 and Item 2

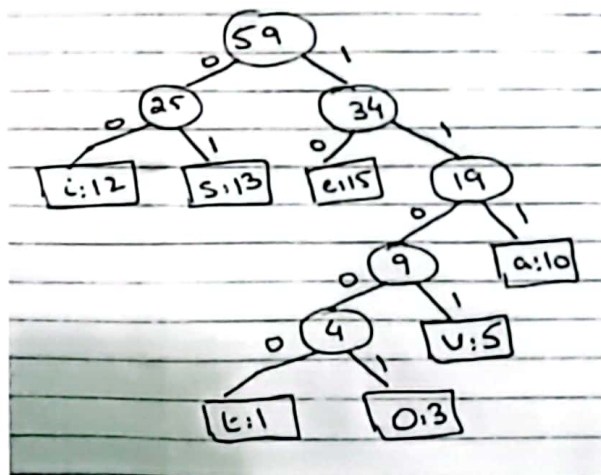
Question 5 Greedy Algorithms [10 Marks]

A file contains the following characters with the frequencies as shown. If Huffman coding is used for data compression, determine the following:

Characters	Frequencies
A	10
E	15
I	12
O	3
U	5
S	13
T	1

11

Question 5a: Draw the Huffman tree [6 points].



Question 5b: What is the Huffman code for each character. Write in the table below [2 points]

Characters	Huffman Code
A	111
E	10
I	00
O	11001
U	1101
S	01
T	11000

Question 5c: Decode the following message 01 111 11000 using the Huffman tree generated above. [2 points]
SAT

Question 6 Algorithms Design [10 Marks]

Consider an array $A[1 \dots n]$ of integers. A number a is a heavy hitter (majority element) in A if a occurs in A at least $n/2$ times. Give an efficient algorithm (linear time) that finds the heavy hitter in a given array A and write the pseudocode for it. The Array A will always contain a majority element

PS. You should use $O(1)$ storage space to get full credit.

Initially, memory = empty and counter = 0.

When element a_t arrives

```
if (counter == 0)
    set memory =  $a_t$  and counter = 1
else
    if  $a_t == \text{memory}$ 
        counter++
    else
        counter--
        discard  $a_t$ 
```
