National University of Computer and Emerging Sciences, Lahore Campus

Course:	Design & Analysis of Algorithms	Course Code:	CS2009
Program:	BS (Computer Science)	Semester:	Spring 2023
Duration:	20 Minutes	Total Marks:	15
Paper Date:	21-Feb-2023	Weight	2.5
Section:	J	Page(s):	2
Exam:	Quiz 1	Reg. No.	[redacted]

Instruction/Notes:

11/15

Question 1: [5 marks]

For the following functions f(n) and g(n), indicate whether f(n) = O(g(n)), f(n) = Ω(g(n)), or both, i.e. f(n) = θ(g(n)). Justify your answer.

$f(n) = n^2 \log n$

$g(n) = 2^{1/2} n^2 \log(2^{1/2} n)$

Assuming we are taking log base of 2

$O(n) \Rightarrow f(n) \le c \cdot g(n)$

$n^2 \log n \le c \cdot 2^{1/2} n^2 \log(2^{1/2} n)$

$\log n \le c \cdot 2^{1/2} \log(2^{1/2} n)$

$\frac{\log n}{\sqrt{2} \log(\sqrt{2} n)} \le c$

assume if $c = 3$

$\frac{\log n}{\sqrt{2} \log_2(\sqrt{2} n)} \le 3 \Rightarrow$ if $n_0 = 10$

$\frac{\log_2 10}{\sqrt{2} \log_2(\sqrt{2} \times 10)} = 1.202$

$0.615 \le 3$

$\Omega(n) \Rightarrow c \cdot g(n) \le f(n)$

$c \cdot 2^{1/2} n^2 \log(2^{1/2} n) \le n^2 \log n$

$c \cdot 2^{1/2} \log(\sqrt{2} n) \le \log n$

$c \le \frac{\log n}{\sqrt{2} \log(\sqrt{2} n)}$

Assume $c = 0.5$

$0.5 \le \frac{\log n}{\sqrt{2} \log(\sqrt{2} n)}$

if $n_0 = 5$

$0.5 \le 0.581$

Therefore $O(n)$ & $\Omega(n)$ justifying.

Question 2: [5 Marks]

Find big-theta of the function $f(n) = n/18 - 19n^{1/2} + 20$, give the constants c_1, c_2, n_0

$O(n) = n/18 - 19n^{1/2} + 20$

$f(n) \le c \cdot g(n)$

$n/18 - 19n^{1/2} + 20 \le c \cdot n$

Assume $c = 10$

if $n_0 = 15$

$15/18 - 19(15)^{1/2} + 20 \le 10(15)$

$-52.8 \le 150$

if $\Omega(n) \Rightarrow c \cdot g(n) \le f(n)$

$c \cdot n \le n/18 - 19n^{1/2} +$

3/5

Question 3: [5 marks]

Write down the Running Time equation (T(n)) of the following algorithm and analyze its time complexity. Show complete steps. If you make any assumptions, state them clearly.

```
int algo(input, n)
{
If (n<=0)
        {return 0}
X = algo(A, n/2)
Y = algo(A, n/4)
Z = A[(n/2) + (n/4) +1]
return (X+Y+Z)
}
```

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + cn$$

Levels = k

Total work = $cn(\log n + 1)$

$= n \log n$

$= O(n \log n)$

Time Complexity = $\frac{n}{2^k} = 1$

$\log_2 n + 1$

3/5

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**6-Feb-18**	**Weight**	**4**
Section:	**C**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 1**	**Section:**	

Write down the recurrence relation for the following algorithm and analyze its time complexity using iteration method. Show complete steps

```
SORT(A,n){
    if n > 1{
       SORT(A,n-1)
       key = A[n]
       i = n-1
       while i > 0 and A[i] > key
             A[i+1] = A[i]
             i = i - 1
       A[i + 1] = key
      }
}
```

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**6-Feb-18**	**Weight**	**4**
Section:	**D**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 1**	**Section:**	

Write down the recurrence relation for the following algorithm and analyze its time complexity using iteration method. Show complete steps

```
sort(A, n){

  if(n>1{
    for i =1 to n
       if (A[i] > A[i +1]) then
          swap (A[i], A[i+1])

    sort(A, n-1)
}

}
```

Design and Analysis of Algorithms (CS302) Section 6C
Quiz 1

Q1) True / False: Justify your answer (10 Marks = 2.5 %)

1. $3^{n+4} = O(3^n)$

2. $2^{6n} = O(2^n)$

3. 9^n is $\Theta(3^n)$

4. 8^n is $\Omega(4^n)$

5. $3n^2+20n$ is $\Theta(n^3)$

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**6-Feb-18**	**Weight**	**4**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 1(a)**	**Section:**	

True/False Justify your answer

1. $3^{4n} = \Theta(6^{4n})$ False as $6^{4n} = 2^{4n}3^{4n}$ so can't be the lower bound
2. $3^{4n} = O(6^{4n})$ True
3. $3^{4n} = \Omega(6^{4n})$ False
4. $\lg n = \Theta(\ln n)$ True changing base of log does not change asymptotic bounds
5. $\lg n = \Omega(\ln n)$ True

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**6-Feb-18**	**Weight**	**4**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 1(b)**	**Section:**	

True/False Justify your answer

1. $n^2 = \Theta(4^{\lg n})$ True as $4^{\lg n} = 2^{2\lg n} = 2^{\lg n^2} = n^2$
2. $n^2 = O(4^{\lg n})$ True
3. $n^2 = \Omega(4^{\lg n})$ True
4. $n\lg n = \Theta(\ln n^n)$ True since change in log base does not change asymptotic bounds
5. $n\lg n = \Omega(\ln n!)$ True as $\lg n! = \Theta(n\lg n)$

National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Design and Analysis of Algorithms	Course Code:	CS302
	Degree Program:	BS(CS)	Semester:	Spring 2020
	Duration:	20 Minutes	Total Marks:	4+3+3
	Date:	Feb 21, 2020	Weight	2.5
	Section:	6C	Page(s):	1
	Exam Type:	Quiz		

Student : Name:_____ Roll No._____
Section:_____

Instruction/ Notes: Attempt the quiz on this sheet and write concise answers.

Q1) You are given a 2D sorted array (elements are in ascending order in each row and column) of size NxN. Your task is to devise a *linear time* algorithm to find an element *x* in the 2D array.

Q2) Assume that $T(1)=1, T(2)=1$, and $T(n)=T(n-2)+n \; for \; k \geq 3$. What is $T(n)$ in terms of big O notation?

Q3) Justify your answer whether the following recurrences can be solved using Master theorem. **Note**: Show your working on the other side of the page.

1. $T(n)=3T\left(\frac{n}{3}\right)+n\,lg\,n$
2. $T(n)=2T(\frac{n}{4})+\sqrt{n}$
3. $T(n)=T(n-2)+n^2$

National University of Computer and Emerging Sciences, Lahore Campus

	Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
	Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
	Duration:	**10 Minutes**	**Total Marks:**	**10**
	Paper Date:	**20-Feb-18**	**Weight**	**3**
	Section:	**C**	**Page(s):**	**1**
			Roll No:	
	Exam:	**Quiz 2**	**Section:**	

You are given an array A of size n and an integer k such that the sub arrays A[1.. n/k], A[n/k+1..2n/k] … A[(k-1)n/k+1 .. n] are sorted. Write the function k-merge that sorts the entire array A.

National University of Computer and Emerging Sciences, Lahore Campus

	Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
	Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
	Duration:	**10 Minutes**	**Total Marks:**	**10**
	Paper Date:	**20-Feb-18**	**Weight**	**3**
	Section:	**D**	**Page(s):**	**1**
			Roll No:	
	Exam:	**Quiz 2**	**Section:**	

Suppose you are given k sorted arrays, each with n elements, and you want to combine them into a single array of kn elements. Consider the following approach. Using the merge subroutine taught in lecture, you merge the first 2 arrays, then merge the $3rd$ given array with this merged version of the first two arrays, then merge the $4th$ given array with the merged version of the first three arrays, and so on until you merge in the final (kth) input array. What is the running time taken by this successive merging algorithm, as a function of k and n? Show complete working.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**20-Feb-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 2(a)**	**Section:**	

Below is the code for selection sort. Draw the decision tree of selection sort when sorting 3 elements

```
SelectSort(A,n)
{
        for(i=1 to n-1){
                min = i
                for(j=i+1 to n){
                        if(A[j]<A[min])
                                min=j
                }
                swap(A[min],A[i])
}
```

Decision tree for the selection sort of {a, b, c}.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**20-Feb-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 2(b)**	**Section:**	

Below is the code for Bubble sort. Draw the decision tree of Bubble sort when sorting 3 elements

```
BubbleSort(A,n)
{
        for(i=1 to n-1){

                for(j= 1 to n-i){
                        if(A[j+1]<A[j])
                                swap(A[j],A[j+1])
                }

}
```

Let V1, V2 and V3 be the initial unsorted elements, the decision tree would be

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**15-March-18**	**Weight**	**3**
Section:	**C**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 3**	**Section:**	

We have derived the worst case height of a B-tree of order 2t in class. What would be the best case height of a B-tree of order 2t? Derive the height as a function of number of data items in the tree (n). Also show complete working.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**15-March-18**	**Weight**	**3**
Section:	**D**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 3**	**Section:**	

Suppose you want to implement B-trees for indexing in a database. The block size of the disk is 1024 bytes, each data value and pointer pair is of 16 bytes, child pointer is of 4 bytes. Each node in the B-tree has a 4 byte integer to store the number of data items in the node and a 1 byte Boolean variable for isLeaf attribute. Given this information what should be the best suitable value of t? Show complete working for your answer.

As node size of B-tree must not exceed the block size, following relation must hold

$16(2t-1)+4(2t)+4+1<=1024$

$32t-16+8t+4+1<=1024$

$40t<=1035$

So $t = 25$

Name: _____ Reg #: _____ Section: _____

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS2009**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2022**
Duration:	**30 Minutes**	**Total Marks:**	**15**
Paper Date:	**28-April-2022**	**Weight**	**%**
Section:	**4B**	**Page(s):**	**4**
Exam:	**Quiz 3**		

Q1) The semester is over! You've rented a car and are ready to set out on a long drive on the Strange Highway. There are n tourist stores on the Strange Highway, numbered 1, 2, …, n and you would want to stop at these and buy some souvenirs (only one souvenir may be bought from a store and all souvenirs everywhere have the same price). You are a greedy shopper and are most interested in maximizing the total discount you get on your shopping. You know that each store i offers a discount d_i on a souvenir. But it's a strange highway after all. It turns out that you can only buy a souvenir from a store i if you have not bought anything from the previous f_i stores. For example, if $f_6 = 3$ then you can only buy a souvenir from store 6, and get the discount d_6, if you haven't bought anything from stores 3, 4, and 5. All the d_i and f_i are known to you in advance (passed as input). You have recently learnt the DP technique in your algorithms course and wish to apply it here in order to maximize your total discount under the given constraints. I will help you by defining the optimal sub-structure. [6 Marks]

$D[i]$ = max total discount for the trip *till store* i whether buying at store i or not.

Example

i	f(i)	d(i)	D(i)
1	0	2	2
2	0	4	6
3	2	2	6
4	2	5	7

(a) Provide recurrences for D[i].

$$D[i] = \max(D[i - f_i - 1] + d_i, D[i - 1])$$

(b) Provide complete pseudo-code of the bottom-up DP algorithm based on the recurrence above.

```
int Function( char *d, char *f, int n )
{
   D[1] = d[1]
   for (i = 2 to n)
         D[i] = max (D[i-1], D[i – f[i]-1] +d[i])

  return D[n]
}
```

Name: _____ Reg #: _____ Section: _____

Q2) Consider the following recursive algorithm. [2 + 5+ 2 = 9 Marks]

```
/* m and n are lengths of char arrays X and Y respectively */
int Function( char *X, char *Y, int m, int n )
{
   if (m == 0 || n == 0)
      return n+m;
   if (X[m-1] == Y[n-1])
      return 1 + Function (X, Y, m-1, n-1);
   else
      return min (Function (X, Y, m, n-1) +1,  Function (X, Y, m-1, n) +1);
}
```

(a) What is time complexity of above algorithm?

$O(n^2)$ or $O(m^2)$

(b) Convert the recursive code given above into bottom up iterative dynamic programming algorithm.

```
int Function( char *X, char *Y, int m, int n )
{
   for (i = 1 to n)
         A[i][0] = i
    for (j = 1 to m)
         A[0][j] = j

   for (i = 1 to n)
            for (j = 1 to m)
                  if (X[m-1] == Y[n-1])
                        A[i][j] = A[i-1][j-1] +1
                  Else
                        A[i][j] = min (A[i][j-1] , A[i-1][j])

return A[m][n]
}
```

(c) What is time complexity of your iterative algorithm?

O(m*n)

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**15-March-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 3(a)**	**Section:**	

Given the following B-tree where t=3, insert the following keys in order 14, 38, 18 and 25. Show the tree after each subsequent insertion.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**15-March-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 3(b)**	**Section:**	

Given the following B-tree where t=3, insert the following keys in order 11, 45, 16 and 9. Show the tree after each subsequent insertion.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**3-April-18**	**Weight**	**3**
Section:	**C**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 4**	**Section:**	

What is an optimal Huffman code for the following set of frequencies, based on the first 8 Fibonacci numbers?
a:1 b:1 c:2 d:3 e:5 f:8 g:13 h:21
Can you generalize your answer to find the optimal code when the frequencies are the first n Fibonacci numbers?

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**3-April-18**	**Weight**	**3**
Section:	**D**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 4**	**Section:**	

Suppose that in activity selection problem, instead of always selecting the first activity to finish, we instead select the last activity to start that is compatible with all previously selected activities. Is this a greedy strategy? Will it always yield the optimal answer?

Name: _____ Reg #: _____ Section: _____

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS2009**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2022**
Duration:	**30 Minutes**	**Total Marks:**	**20**
Paper Date:	**1-June-2022**	**Weight**	**%**
Section:	**4B**	**Page(s):**	**3**
Exam:	**Quiz 4**		

Q1) Given a weighted directed graph in adjacency list representation, write an efficient algorithm to calculate in-degree (number of incoming edges), out-degree (number of outgoing edges), sum of weight of incoming edges, and sum of weight of outgoing edges for each vertex of the graph. Analyze time complexity of your algorithm. [10 Marks]

Solution

Run BFS and use 4 different arrays of size n (n = total vertices) to store in-degree (number of incoming edges), out-degree (number of outgoing edges), sum of weight of incoming edges, and sum of weight of outgoing edges for each vertex.

BFS(G, start)

```
Create new queue Q
Q.push(start)
color[1..n] = White

while Q is not empty
    u = Q.pop()
    for each node v adjacent to u
        if color[v] = White then
            color[v] = Black
            indegree[v]++
            outdegree[u]++
            sumOfIndegree[v] += weight(u,v)
            sumOfOutdegree[u] += weight(u,v)
            Q.push(v)
```

Time complexity = O(V+E)

Q2) You are given a weighted undirected graph G(V, E) and its MST T(V, E'). Now suppose an edge (a, b) ϵ E' has been deleted from the graph. You need to devise an algorithm to update the MST after deletion of (a, b).

Describe an algorithm for updating the MST of a graph when an edge (a, b) is deleted from the MST (and the underlying graph). It's time complexity must be better than running an MST algorithm from scratch. State and explain the time complexity of your algorithm. Analyze time complexity of your algorithm.

You can assume that all edge weights are distinct, that the graph has E edges and V vertices after the deletion, that the graph is still connected after the deletion of the edge, and that your graph and your MST are represented using adjacency lists. [10 Marks]

Solution

The following algorithm has time complexity O(E). Run DFS twice on what remains of the MST, once starting on a and once starting on b to determine the two connected components A and B, which you store in two HashSets. This takes O(E). Then iterate through all edges going out from the elements of A. If an edge leads to an element of B (can be check in O(1) using HashSet), check whether it is the cheapest edge so far. If it is cheaper than any other edge so far, store it. So, finding the cheapest edge can also be done in O(E). In the end, add the cheapest edge to the MST.

Name: _____ Reg #: _____ Section: _____

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS2009**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2022**
Duration:	**30 Minutes**	**Total Marks:**	**20**
Paper Date:	**1-June-2022**	**Weight**	**%**
Section:	**4B**	**Page(s):**	**3**
Exam:	**Quiz 4**		

Q1) Given a weighted directed graph in adjacency list representation, write an efficient algorithm to calculate in-degree (number of incoming edges), out-degree (number of outgoing edges), sum of weight of incoming edges, and sum of weight of outgoing edges for each vertex of the graph. Analyze time complexity of your algorithm. [10 Marks]

Solution

Run BFS and use 4 different arrays of size n (n = total vertices) to store in-degree (number of incoming edges), out-degree (number of outgoing edges), sum of weight of incoming edges, and sum of weight of outgoing edges for each vertex.

BFS(G, start)

```
Create new queue Q
Q.push(start)
color[1..n] = White

while Q is not empty
    u = Q.pop()
    for each node v adjacent to u
        if color[v] = White then
            color[v] = Black
            indegree[v]++
            outdegree[u]++
            sumOfIndegree[v] += weight(u,v)
            sumOfOutdegree[u] += weight(u,v)
            Q.push(v)
```

Time complexity = O(V+E)

Q2) You are given a weighted undirected graph G(V, E) and its MST T(V, E'). Now suppose an edge (a, b) ϵ E' has been deleted from the graph. You need to devise an algorithm to update the MST after deletion of (a, b).

Describe an algorithm for updating the MST of a graph when an edge (a, b) is deleted from the MST (and the underlying graph). It's time complexity must be better than running an MST algorithm from scratch. State and explain the time complexity of your algorithm. Analyze time complexity of your algorithm.

You can assume that all edge weights are distinct, that the graph has E edges and V vertices after the deletion, that the graph is still connected after the deletion of the edge, and that your graph and your MST are represented using adjacency lists. [10 Marks]

Solution

The following algorithm has time complexity O(E). Run DFS twice on what remains of the MST, once starting on a and once starting on b to determine the two connected components A and B, which you store in two HashSets. This takes O(E). Then iterate through all edges going out from the elements of A. If an edge leads to an element of B (can be check in O(1) using HashSet), check whether it is the cheapest edge so far. If it is cheaper than any other edge so far, store it. So, finding the cheapest edge can also be done in O(E). In the end, add the cheapest edge to the MST.

Name: _____ Reg #: _____ Section: _____

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS2009**
Program:	**BS(Computer Science)**	**Semester:**	**Fall 2022**
Duration:	**20 Minutes**	**Total Marks:**	**10**
Paper Date:	**28-Nov-2022**	**Weight**	**%**
Section:	**5A**	**Page(s):**	**3**
Exam:	**Quiz 4**		

Q1) Given a directed graph, find the total number of routes to reach the destination from a given source with exactly m edges. For example, consider the following graph: [10 Marks]

Let source = 0, destination = 3, number of edges m = 4. The graph has 3 routes from source 0 to destination 3 with 4 edges. The solution should return the total number of routes 3. Give an efficient algorithm for solving this problem and analyze its time complexity.

0 —> 1 —> 5 —> 2 —> 3
0 —> 1 —> 6 —> 5 —> 3
0 —> 6 —> 5 —> 2 —> 3

Solution:

The idea is to do a BFS traversal from the given source vertex. BFS is generally used to find the shortest paths in graphs/matrices, but we can modify normal BFS to meet our requirements. Usually, BFS doesn't explore already discovered vertices again, but here we do the opposite. To cover all possible paths from source to destination, remove this check from BFS. But if the graph contains a cycle, removing this check will cause the program to go into an infinite loop. We can easily handle that if we don't consider nodes having a BFS depth of more than `m`. Basically, we maintain two things in the BFS queue node:

- The current vertex number.
- The current depth of BFS (i.e., how far away from the current node is from the source?).

So, whenever the destination vertex is reached and BFS depth is equal to `m`, we update the result. The BFS will terminate when we have explored every path in the given graph or BFS depth exceeds `m`.

```
int findTotalPaths(Graph const &graph, int src, int dest, int m)
{
    // create a queue for doing BFS
    queue<Node> q;

    // enqueue source vertex
    q.push({src, 0});

    // stores number of paths from source to destination having exactly `m` edges
    int count = 0;

    // loop till queue is empty
    while (!q.empty())
    {
        // dequeue front node
        Node node = q.front();
        q.pop();

        int v = node.vertex;
        int depth = node.depth;

        // if the destination is reached and BFS depth is equal to `m`, update count
        if (v == dest && depth == m) {
            count++;
        }

        // don't consider nodes having a BFS depth more than `m`.
        // This check will result in optimized code and handle cycles
        // in the graph (otherwise, the loop will never break)
        if (depth > m) {
            break;
        }

        // do for every adjacent vertex `u` of `v`
        for (int u: graph.adjList[v])
        {
            // enqueue every vertex (discovered or undiscovered)
            q.push({u, depth + 1});
        }
    }

    // return number of paths from source to destination
    return count;
}
```

Time complexity of your algorithm:

Name: _____ Reg #: _____ Section: _____

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS2009**
Program:	**BS(Computer Science)**	**Semester:**	**Fall 2022**
Duration:	**20 Minutes**	**Total Marks:**	**10**
Paper Date:	**28-Nov-2022**	**Weight**	**%**
Section:	**5C**	**Page(s):**	**2**
Exam:	**Quiz 4**		

Q1) Given a weighted directed graph in adjacency list representation, write an efficient algorithm to calculate in-degree (number of incoming edges), out-degree (number of outgoing edges), sum of weight of incoming edges, and sum of weight of outgoing edges for each vertex of the graph. Analyze time complexity of your algorithm. [10 Marks]

Solution

Run BFS and use 4 different arrays of size n (n = total vertices) to store in-degree (number of incoming edges), out-degree (number of outgoing edges), sum of weight of incoming edges, and sum of weight of outgoing edges for each vertex.

```
BFS(G, start)
   Create new queue Q
   Q.push(start)
   color[1..n] = White

   while Q is not empty
      u = Q.pop()
      for each node v adjacent to u
         if color[v] = White then
            color[v] = Black
            indegree[v]++
            outdegree[u]++
            sumOfIndegree[v] += weight(u,v)
            sumOfOutdegree[u] += weight(u,v)
            Q.push(v)
```

Time complexity = O(V+E)

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**3-April-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 4(a)**	**Section:**	

Suppose that in binary knapsack problems, all the items have unit weight but different value. Given set ***I***, of ***n*** items and bag capacity ***W***, devise an algorithm that select a subset of items whose collective weight does not exceede the capacity of bag, yet maximize the collective profit of selected items. Describe your algorithm in words. Is this algorithm a dynamic programming algorithm?

As all the items have unit weight that means we can always put the W number of items in the bag for some specified capacity W. So collective profit will be maximize by the greedy strategy i.e. select that item first whose value is maximum.

Sort all the items according to the value

While capacity is not full add item i and update the bag remaining capacity

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**3-April-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 4(b)**	**Section:**	

Suppose that in binary knapsack problems, all the items have unit value but different weight. Given set ***I***, of ***n*** items and bag capacity ***W***, devise an algorithm that select a subset of items whose collective weight does not exceede the capacity of bag, yet maximize the collective profit of selected items. Describe your algorithm in words. Is this algorithm a dynamic programming algorithm?

As all the items have unit value that means maximizing the number of items in the bag will maximize the collective profit. So collective profit will be maximize by the greedy strategy i.e. select that item first whose weight is minimum.

Sort all the items according to the weight

While capacity is not full add item i and update the bag remaining capacity

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**26-April-18**	**Weight**	**3**
Section:	**C**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 5**	**Section:**	

Proof or disproof that for any directed graph G, transpose of the component graph of G^T is the same as the component graph of G.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**26-April-18**	**Weight**	**3**
Section:	**D**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 5**	**Section:**	

Give an O(V + E) time algorithm to compute the component graph of a directed graph G (V,E). Make sure that there is at most one edge between two vertices in the component graph your algorithm produces.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**26-April-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 5(a)**	**Section:**	

Assume that G(V,E) is a weighted undirected graph where edge weights may not be distinct. Either proof or disproof the following statement.

Suppose each edge weight is increased by 1 in the graph G. The MST (set of tree edges) of changed graph would be same as the MST of the original graph.

If it is correct give the argument/justification or give a counter example if it is incorrect.

Yes as the relative order of the edge weights remain same so Kruskal Algorithm will select the same set of edges.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**26-April-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 5(b)**	**Section:**	

Assume that G(V,E) is a weighted undirected graph where edge weights may not be distinct. Either proof or disproof the following statement.

If G has more than |V|-1 edges and there is a unique edge with highest weight, then this edge cannot be part of minimum spanning tree (MST).

If it is correct give the argument/justification or give a counter example if it is incorrect.

False the maximum weight edge can be part of the MST if that edge is a bridge e.g in the graph below if edge 0,3 has maximum weight then it will always be a part of MST

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**8-May-18**	**Weight**	**3**
Section:	**C**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 6**	**Section:**	

Given a weighted, directed graph G (V, E) with no negative-weight cycles, let m be the maximum number of edges in the shortest path from source s to v for all vertices v in V. (Here, the shortest path is by weight, not the number of edges.) Suggest a simple change to the Bellman-Ford algorithm that allows it to terminate in m + 1 passes i.e its time complexity should be O(mV), even if m is not known in advance.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**8-May-18**	**Weight**	**3**
Section:	**D**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 6**	**Section:**	

The following is an algorithm for finding single source shortest-paths for a graph G with weights on the edges, that can be negative (Assume that there is no negative weight cycle). Does it solve the problem?

a. Find the minimal weight in G, W_{min}
b. To each weight in G, add $|W_{min}|$ (absolute value of W_{min})
c. Execute dijkstra algorithm

Justify your answer.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**8-May-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 6(a)**	**Section:**	

Suppose we change the line while (Q is not empty) with while (|Q| >1) of Dijkstra's algorithm.

This change causes the **while** loop to execute |V| - 1 times instead of |V | times. Is this proposed algorithm correct? Justify your answer.

National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 2018**
Duration:	**10 Minutes**	**Total Marks:**	**10**
Paper Date:	**8-May-18**	**Weight**	**3**
Section:	**E**	**Page(s):**	**1**
		Roll No:	
Exam:	**Quiz 6(b)**	**Section:**	

Professor Gaedel has written a program that he claims implements Dijkstra's algorithm. The program produces values of *d* and π for each vertex v in V. Give an O(V +E) time algorithm to check the output of the professor's program. Given the graph G, d and π, it should determine whether the d and π attributes match those of some shortest-paths tree of G (i.e whether they are correct or not). You may assume that all edge weights are nonnegative. Explain your algorithm in 2-3 lines.

Quiz-1 **Algorithms section B, FALL 2016**

Q-1: Given mergesort, at each level j=0,1,2..logn, there are _____ subproblems each of size _____. (recall the recursion tree of merge sort)
(2)

Q-2: Give the time complexity T(n) of MERGE routine in merge sort. Show how it is a linear equation. **(4)**

Assume that from p to r, there are n numbers.

Q-3: Below is the pseudocode of bubblesort:
(4)

```
BUBBLESORT(A)
for i ← 1 to length[A]
    do for j ← length[A] downto i + 1
        do if A[j] < A[j − 1]
            then exchange A[j] ↔ A[j − 1]
```

Show each iteration of the pseudocode on an input array of [4 6 1 5 2]. Clearly show the pointers i and j at each iteration.

Quiz 2 **Algorithm Section B – FALL 2016**

Q 1: Find time complexity of following recurrences using Master method. Mention which case applies and show working.
(5)

1) $T(n) = 3T(n/2) + n$
2) $T(n) = 0.5T(n/2) + 1/n$
3) $T(n) = 16T(n/4) + n!$
4) $T(n) = 3T(n/3) + 5n$
5) $T(n) = 4T(n/2) + n^2$

Q 2: Write upper bound, Big-Oh, O() of following time complexities.
(2)

$T(n) = 10n^3 + 4n^2 + 16n$
$T(n) = n\log n + n^2 + 100$

Q 3: Write an algorithm (using C/C++/Java or pseudocode as shown in class) to **find 2nd largest element in an unsorted array**. Give time complexity of your algorithm. Don't use any built in functions. To swap 2 numbers you may use swap(i, j) method. Method should return the 2nd largest element. (5)

```
int Find2ndMax(Arr) {}
```

Object Oriented Programming (C&D) – Spring 2020
Quiz-03

Implement following **ComplexNumber** class:

```
class ComplexNumber
{
private:
   int real;
   int imaginary;
public:
   ComplexNumber();
   ComplexNumber(int , int ) ;
   ComplexNumber(const ComplexNumber
);
   ~ComplexNumber(); //Does Nothing.

   void Input();
   void Output();

   bool IsEqual(ComplexNumber);
   ComplexNumber Conjugate();
   ComplexNumber Mult(ComplexNumber);
   float Magnitude();
};
```

Sample Run:

```
C:\Windows\system32\cmd.exe
Enter c1:
Enter Real:     2
Enter Imanginary:       3


Enter c2:
Enter Real:     4
Enter Imanginary:       5

c1 = 2+3i

c2 = 4+5i

c1 is NOT Equal to c2.

Conjugate of c1:        2-3i

Conjugate of c2:        4-5i

c1 + c2 :       6+8i

c1 - c2 :       -2-2i

c1 x c2 :       Do Yourself

Magnitude of c1 = Display Magnitude of c1 here


Magnitude of c2 = Display Magnitude of c2 here
Press any key to continue . . .
```

Important:

- Do not change class definition
- Submit only one **RUNNING** file YourRollNumber.cpp that contains class, its implementation and the driver Program. Do not submit .rar or .zip files.

Quiz 3 **Algorithm Section B – FALL 2016**

Q 1: Assume that you want to go to from FAST to railway station and it will take following amount of time on a particular route. FAST to kalma chowk takes 10 minutes, FAST to canal road takes 4 minutes, Kalma chowk to Mall road takes 6 minutes, Canal to Mall rd takes 7, Mall to station in 5 minutes, Canal to station via scheme more takes 14 minutes.

a. How can you represent this as a graph? (3)
b. What is the shortest path from FAST to railway station? Show the path clearly and total minutes taken by you path. (3)
c. Which algorithm will you use for this calculation? (1) Bonus

Q 2: What is the space requirements of adjacency list? Write in terms of V and E.

Q 3: What is the space requirements of adjacency matrix? Write in terms of V and E.

Quiz 3 **Algorithm Section B – FALL 2016**

Q 1: Assume that you want to go to from FAST to railway station and it will take following amount of time on a particular route. FAST to kalma chowk takes 10 minutes, FAST to canal road takes 4 minutes, Kalma chowk to Mall road takes 6 minutes, Canal to Mall rd takes 10, Mall to station in 5 minutes, Canal to station via scheme more takes 14 minutes.

a. How can you represent this as a graph? (3)
b. What is the shortest path from FAST to railway station? Show the path clearly and total minutes taken by you path. (3)
c. Which algorithm will you use for this calculation? (1) Bonus

Q 2: What is the space requirements of adjacency matrix? Write in terms of V and E.

Q 3: What is the space requirements of adjacency list? Write in terms of V and E.

Quiz 4

Design and analysis of algorithms - Section B

Q1: a) Draw a directed acyclic graph (DAG) having atleast 6 vertices and it should not be sparse. (2)

b) Suppose the graph you have made represents some activities, how will you order your activities?

Show the ordering and tell in one line which strategy did you use? (3+3)

c) Which is the last activity (activities) you can do? (1)

d) Which is the first activity (activities) you can do? (1)

Quiz 5

Analysis of algorithms

Q 1.a: Given that all coins and currency notes in Pakistan are available with infinite supply, how will you find change of an amount of money such that number of coin/notes used is minimum? Explain your algo/pseudocode in 3 lines and take an example number above 400 and run you algorithm on it. (5)

b) Your algorithm is of which type? (Greedy, Dynamic Prog, Divide n Conquer, Brute force etc). (2)

b) Will your solution work on any given input number (in Pak currency)? (1)

c) Will your solution work on any arbitrary currency (not Pakistani currency but any possible currency)? (2)

Explain b, c, d in 1 line.

Quiz 6: Algorithms-B

December 7, 2016

Q 1: Prime number is a number which is divisible only by 1 and itself. Write a code to check if a number is prime or not.

```
IsPrime(int num){}
```

Q 2: Given 2 strings, a and b, find if string b is a substring of a.

```
ContainsStr(str a, str b) {}
```

Code both in c++, without using any library function.

Check correctness of your code after taking sample input and dry running on it.

What will be the complexity of both 1) and 2)?

National University of Computer and Emerging Sciences, Lahore Campus

Course:	Design and Analysis of Algorithms	Course Code:	
Program:	BS (Software Engineering)	Semester:	Spring 2025
Duration:	15 Minutes	Total Marks:	10
Paper Date:	13-Feb-2025	Weight	
Section:	BSE-6A	Page(s):	01
Exam:	Quiz 1 Version 2	Roll. No.	22L-7983
		Name:	Imran Ahmad
Instruction/Notes:	Honesty always gives fruit and Dishonesty is always harmful.		

5/10

Q1: Derive the recurrence of following recursive algorithms. Please mention at each step and then give final equation.

```
Mystery (N)
    If (N > 1)
        Then  Print "Deriving recurrence is fun"
              Mystery (2N/3)
              for (i=1 to N)
                    Print "Solving recurrence is fun"
              Mystery (N/5)
```

$T(n) = T(2N/3) + T(N/5) + \theta(N)$

5)

Insertion Sort

Q2: A tournament system records player scores throughout a competition. The leaderboard needs to remain sorted at all times as new scores are added. The total number of players is relatively small, but updates occur frequently. The goal is to maintain an updated ranking efficiently.

Which sorting algorithm is the best choice explain the reason with time and space complexity?

PTO

National University of Computer and Emerging Sciences, Lahore Campus

Course:	Design and Analysis of Algorithms	Course Code:	
Program:	BS (Software Engineering)	Semester:	Spring 2025
Duration:	30 Minutes	Total Marks:	10
Paper Date:		Weight	02
Section:	BSE-6C	Page(s):	
		Roll. No.	22L-7992
Exam:	Quiz 3 Version 2	Name:	Waseem

Instruction/Notes: Honesty always gives fruit and Dishonesty is always harmful.

1. Which sorting algorithm is best suited for sorting large numbers with a fixed digit length?
   a) Count Sort
   b) Quick Sort
   c) Radix Sort
   d) Bucket Sort

2. Which sorting algorithm is NOT stable by default?
   a) Bucket Sort
   b) Quick Sort
   c) Counting Sort
   d) Radix Sort

3. What is the worst-case time complexity of Quick Sort?
   a) O(nlogn)
   b) O(n^2)
   c) O(n)
   d) O(logn)

4. Bucket Sort is most effective when:
   a) The input values are uniformly distributed
   b) The input values are already sorted
   c) The input contains duplicate elements
   d) The input is in reverse order

5. Which sorting algorithm is typically preferred when memory usage is a major constraint?
   a) Bucket Sort
   b) Quick Sort
   c) Counting Sort
   d) Radix Sort

6. If Heap Sort takes 1 second for an input of size 100,000, how long will it take for an input of size 1,000,000?
   a) 2 seconds
   b) 10 seconds
   c) 20 seconds
   d) 50 seconds

7. In a comparison sort, how many comparisons are needed in the worst case to sort 10,000 elements?
   a) O(10^4)
   b) O(10^4log10^4)
   c) O(10^8)
   d) O(log10^4)

8. Radix Sort sorts numbers using Counting Sort as a subroutine. If we have 1 million numbers, each with 5 digits, how many times will Counting Sort run?
   a) 1
   b) 5
   c) 10
   d) 10

9. If a dataset contains 50,000 numbers in the range 1 to 500,000, which sorting algorithm will perform best?
   a) Quick Sort

b) Bucket Sort
c) Counting Sort
d) Radix Sort

10. Given an array of 1 million integers where each number is between 1 and 10,000, what is the best sorting approach?
a) Quick Sort
b) Bucket sort
● Counting Sort
d) Radix Sort

11. An array contains 18 elements with values ranging from 100 to 125. How many slots are required in the auxiliary count array for Counting Sort?

●. 18
b). 25
c). 26
d). 27

12. After applying Counting Sort to the array 4,2,2,8,3,3,14, 2, 2, 8, 3, 3, 14,2,2,8,3,3,1, what will be the last two elements of the sorted array?

wrong Question

a. 8,48, 48,4
b. 3,83, 83,8
● 4,84, 84,8
d. 8,38, 38,3

13. for i in range(1, k+1):

C[i] = C[i-1] + C[i]

What is missing in this loop that could cause incorrect sorting?

a. Initialization of C[i] before this loop
● Incrementing C[i] instead of summing
c. Loop should run from 0 to k instead of 1 to k
d. Using C[i−1] instead of C[i+1]

National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Design and Analysis of Algorithms	Course Code:	
Program:	BS(SE)	Semester:	6
Duration:	10 minutes	Total Marks:	12
Paper Date:	20 Mar, 2025	Weight:	
Section:	SE-6A	Page(s):	02
Exam Type:	Quiz3-Version1		

8/12

Student Name: [redacted] Roll No #: [redacted] Section: [redacted]

Instructions: Do Rough Work on the right column.

1) In Counting Sort, which of the following is incorrect regarding its implementation?
a) It does not involve comparisons between elements
b) It modifies the input array in place
c) It requires an auxiliary array of size O(k)
d) It is stable when implemented correctly

2) Why is Radix Sort not commonly used for general-purpose sorting?
a) It is asymptotically slower than Quick Sort
b) It requires additional space and is inefficient for large data
c) It is not stable
d) It does not work for integer sorting

O(d(n+k))

3) If an array contains n elements drawn from a uniform distribution, what is the best way to determine the number of buckets?
a) Use n buckets
b) Use $\sqrt{n}$ buckets
c) Use log(n) buckets
d) Use k buckets

$\frac{100}{10}$ = 10 buckets

4) Bucket Sort is generally inefficient when applied to:
a) Floating-point numbers uniformly distributed between [0,1]
b) Large datasets with arbitrary distributions
c) Sorting 32-bit integers in a small range
d) Sorting data with a normal distribution

O(n²)

5) If k ≫ n (i.e., k is much larger than n), what is the primary issue when applying Counting Sort?
a) It still runs in O(n) time
b) Excessive memory usage
c) Requires additional recursive calls
d) Counting Sort is independent of k

6) What is the worst-case time complexity of Radix Sort for sorting n integers when each integer has d digits and the stable subroutine (Counting Sort) takes O(n + b) time (with b being the base)?
a) O(n)
b) O(d · (n + b))
c) O(n log n)
d) O(n²)

O(d(n+b))

7) Suppose you need to sort an array of strings where each string consists of only lowercase English letters (a-z). What is the most efficient way to use Counting Sort?

a) Sort the strings character by character from right to left using Counting Sort
b) Use Counting Sort directly on the entire string
c) Convert each string into an integer and then use Counting Sort
d) Counting Sort cannot be used for sorting strings

8) You are given an array with n elements, where each element is in the range [0, k]. If $k = O(n^2)$, what is the time complexity of Counting Sort?

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

9) Why must the subroutine in Radix Sort be stable?

a) To handle negative numbers
b) To preserve previous digit orderings
c) To reduce space complexity
d) To improve speed

10) How can Count Sort handle negative integers?

a) Ignore the negative sign
b) Split into positive/negative parts
c) Shift values to make them non-negative
d) Use a hash table

11) A bank needs to sort transactions by 3-digit branch codes (0-999). Which algorithm is best?

a) Radix Sort
b) Bucket Sort
c) Count Sort
d) QuickSort

12) Bucket Sort is generally inefficient when applied to:

a) Floating-point numbers uniformly distributed between [0,1]
b) Large datasets with arbitrary distributions
c) Sorting 32-bit integers in a small range
d) Sorting data with a normal distribution

$O(n+n) = O(n+n^2)$
$O(n^2)$

National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	**Design and Analysis of Algorithms**	**Course Code:**	
Program:	**BS(SE)**	**Semester:**	**6**
Duration:	**20 minutes**	**Total Marks:**	**15 marks**
Paper Date:	**15 May, 2025**	**Weight**	
Section:	**SE-A**	**Page(s):**	2
Exam Type:	**Quiz4-version2**		

Student Name: **Roll No #:**

Instructions: **Solve Question 2 on back side as well**

1) What is the worst-case time complexity of BFS in an adjacency list representation of a graph?
A. $O(V^2)$
B. O(V + E)
C. O(E log V)
D. O(V log V)

2) Which of the following edge types CANNOT exist in DFS of an undirected graph?
A. Tree edge
B. Back edge
C. Forward edge
D. Cross edge

3) What happens if you run Prim's algorithm starting from a vertex with no incident edges?
A. It returns a spanning forest
B. It returns an MST
C. It fails with an error
D. It processes all vertices

4) In a connected graph with n vertices and m edges, how many edges are not in any possible MST?
A. m - (n - 1)
B. Depends on edge weights
C. All edges with maximum weights
D. All edges not on the DFS tree

5) Prim's and Kruskal's algorithms produce the same MST if:
A. The graph is a complete graph
B. All edge weights are equal
C. The graph has unique edge weights
D. The graph is a tree

6) Which is TRUE about the DFS finishing times in a DAG?
A. All sinks finish before sources
B. All sources finish before sinks
C. Sinks finish last
D. Finishing time is irrelevant in DAGs

7) Which of the following statements about DFS is INCORRECT?
A. DFS can be used to detect cycles in a graph
B. DFS can produce topological sorting of a DAG
C. DFS tree in a disconnected graph is always connected
D. DFS discovery and finishing times are essential in articulation point detection

8) Kruskal's algorithm can be best optimized using which technique?
A. Binary Tree Sorting
B. Disjoint Set Union (Union-Find)
C. Breadth First Search
D. Segment Tree

Q2) You are given a grid of cells where each cell is either empty (0) or blocked (1). You start at position (0,0) and want to reach (n-1, m-1). You can move in four directions: up, down, left, right.

Grid:

[0, 0, 0, 0, 1]

[1, 1, 0, 1, 0]

[0, 0, 0, 1, 0]

[0, 1, 1, 0, 0]

[0, 0, 0, 0, 0]

Using DFS, determine the minimum number of steps needed to reach the destination. **Show working as well**

National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	**Design and Analysis of Algorithms**	**Course Code:**	
Program:	**BS(SE)**	**Semester:**	**6**
Duration:	**20 minutes**	**Total Marks:**	**15 marks**
Paper Date:	**15 May, 2025**	**Weight**	
Section:	**SE-A**	**Page(s):**	2
Exam Type:	**Quiz4-version1**		

Student Name: **Roll No #:**

Instructions: **Solve Question 2 on back side as well**

1) Which of the following statements is TRUE regarding the BFS traversal of a directed graph?
A. BFS always visits vertices in lexicographical order
B. BFS tree edges never form a cycle
C. Cross edges in BFS traversal imply a back edge
D. BFS guarantees shortest paths even in weighted graphs

2) Given a graph with negative weight edges but no negative cycles, which traversal is appropriate for finding shortest path from a source?
A. DFS
B. BFS
C. Prim's
D. None of the above

3) Which statement is TRUE about BFS vs DFS?
A. DFS uses more memory than BFS in sparse graphs
B. BFS can get stuck in cycles without visited array
C. DFS is guaranteed to find the shortest path
D. BFS is unsuitable for cycle detection

4) In a connected graph with n vertices and m edges, how many edges are not in any possible MST?
A. m - (n - 1)
B. Depends on edge weights
C. All edges with maximum weights
D. All edges not on the DFS tree

5) Prim's and Kruskal's algorithms produce the same MST if:
A. The graph is a complete graph
B. All edge weights are equal
C. The graph has unique edge weights
D. The graph is a tree

6) Let G be an undirected graph. Suppose we run DFS on G, and classify edges as tree, back, forward, or cross edges. Which of the following must be TRUE?
A. Back edges always form a cycle
B. Cross edges can appear in DFS tree of an undirected graph
C. Every forward edge implies a cycle
D. Cross edges never exist in undirected graphs

7) Which of the following statements about DFS is INCORRECT?
A. DFS can be used to detect cycles in a graph
B. DFS can produce topological sorting of a DAG
C. DFS tree in a disconnected graph is always connected
D. DFS discovery and finishing times are essential in articulation point detection

8) Which of the following guarantees a unique MST in a connected, weighted undirected graph?
A. All edge weights are even numbers
B. All edge weights are unique
C. The graph has no cycles
D. The number of vertices is prime

Q2) You are given a grid of cells where each cell is either empty (0) or blocked (1). You start at position (0,0) and want to reach (n-1, m-1). You can move in four directions: up, down, left, right.

Grid:

[0, 0, 0, 0, 1]

[1, 1, 0, 1, 0]

[0, 0, 0, 1, 0]

[0, 1, 1, 0, 0]

[0, 0, 0, 0, 0]

Using BFS, determine the minimum number of steps needed to reach the destination. **Show working as well**

National University of Computer and Emerging Sciences
Lahore Campus

	Course Name:	Design and Analysis of Algorithms	Course Code:	
	Program:	BS(SE)	Semester:	6
	Duration:	20 minutes	Total Marks:	15
	Paper Date:	15 May, 2025	Weight	
	Section:	SE-6A	Page(s):	
	Exam Type:	Quiz5-version1		

Student Name: **Roll No #:**

Instructions: **Solve Question 2 on back side as well**

1) **What happens if Dijkstra's algorithm is applied to a graph with a negative edge?**
A. It returns the shortest path
B. It returns an incorrect result
C. It crashes
D. It performs slower but correctly

2) **Which of the following is an application of Bellman-Ford?**
A. Topological sorting
B. Cycle detection in undirected graphs
C. Internet routing protocols (e.g., RIP)
D. Minimum spanning tree construction

3) **In Bellman-Ford algorithm, how many times are all edges relaxed in the worst case?**
A. |V|
B. |V| - 1
C. |E|
D. log|V|

4) **Which of the following can be used for shortest paths in a graph with all edge weights equal to 1?**
A. Dijkstra's
B. DFS
C. BFS
D. Kruskal's

5) **In Dijkstra's algorithm, which structure ensures minimum distance vertex is always chosen next?**
A. Stack
B. Queue
C. Min Heap / Priority Queue
D. Set

6) **In a weighted graph, the output of Bellman-Ford is always correct if:**
A. There are no self-loops
B. The graph has no cycles
C. There are no negative weight cycles
D. The graph is a tree

7) **What indicates the presence of a negative weight cycle in Bellman-Ford?**
A. Distances don't change in the final iteration
B. Some distances become zero
C. A vertex is visited more than once
D. Relaxation still changes a distance in the |V|th iteration

8) **What is the major drawback of the Bellman-Ford algorithm compared to Dijkstra?**
A. It doesn't work on sparse graphs
B. It cannot detect negative cycles
C. Its time complexity is higher
D. It only works on trees

```
   5      1
A -----> B ----> C
|      ^      ^
|      |      |
|      |      |
1|     3|      | 6
|      |      |
v      |      |
D -----> E ----> F
   2      1
```

Q2: Start at vertex A and use Dijkstra's algorithm to find the shortest path to all other vertices**. Show each iteration of the algorithm.**

National University of Computer and Emerging Sciences
Lahore Campus

	Course Name:	Design and Analysis of Algorithms	Course Code:	
	Program:	BS(SE)	Semester:	6
	Duration:	20 minutes	Total Marks:	15
	Paper Date:	15 May, 2025	Weight	
	Section:	SE-6A	Page(s):	
	Exam Type:	Quiz5-version2		

Student Name: **Roll No #:**

Instructions: **Solve Question 2 on back side as well**

1) **Which of the following is FALSE about Dijkstra's algorithm?**
A. It can be used to compute SPT
B. It fails with negative edge weights
C. It uses a greedy strategy
D. It can detect negative cycles

2) **In Bellman-Ford algorithm, how many times are all edges relaxed in the worst case?**
A. |V|
B. |V| - 1
C. |E|
D. log|V|

3) **When does Bellman-Ford give better performance than Dijkstra's algorithm?**
A. On sparse graphs with only positive weights
B. On graphs with dense connectivity
C. On graphs with negative weights and sparse structure
D. On graphs with uniform weights

4) **In a weighted graph, the output of Bellman-Ford is always correct if:**
A. There are no self-loops
B. The graph has no cycles
C. There are no negative weight cycles
D. The graph is a tree

5) **Suppose Bellman-Ford runs for |V| iterations and still changes a distance value in the last iteration. What does this indicate?**
A. The graph is a tree
B. The graph is disconnected
C. There exists a negative weight cycle
D. The shortest path hasn't converged

6)) **Which of the following can be used for shortest paths in a graph with all edge weights equal to 1?**
A. Dijkstra's
B. DFS
C. BFS
D. Kruskal's

7) **Which of the following graphs would cause Dijkstra's algorithm to fail to find the correct shortest paths?**
A. Graph with no cycles
B. Graph with disconnected vertices
C. Graph with negative weight edges
D. Graph with uniform weights

8) **You are given a directed graph with negative edge weights but no negative cycles. Which algorithm guarantees correct shortest path results from a single source?**
A. Dijkstra's
B. BFS
C. Bellman-Ford
D. DFS

```
   3      4
A -----> B ----> C
|       ^      ^
|       |      |
|       |      |
2|      5|      | 4
|       |      |
v       |      |
D -----> E ----> F
   1      2
```

Q2: Start at vertex A and use Dijkstra's algorithm to find the shortest path to all other vertices**. Show each iteration of the algorithm.**

National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	**Design and Analysis of Algorithms**	**Course Code:**	
Program:	**BS(SE)**	**Semester:**	**6**
Duration:		**Total Marks:**	**20**
Paper Date:	**14 May, 2025**	**Weight**	
Section:	**SE-B**	**Page(s):**	
Exam Type:	**Quiz4-version2**		

Student Name: **Roll No #:**

Section:

Instructions: **Solve Question 2 on back side and Write Pseudo Code only**

1)**If the heaviest edge in an MST is replaced with a lighter edge not in the original graph, what happens?**
A) The MST remains unchanged
B) A new MST must be formed
C) The graph becomes disconnected
D) It depends on the edge's position

2)**Given edges sorted by weight: (1,2)=2, (2,3)=3, (3,4)=4, (1,4)=5. What is the MST total weight?**
A) 9
B) 10
C) 11
D) 12

3)**What is the worst-case space complexity of BFS in a balanced binary tree with height h?**
A) O(h)
B) O(2^h)
C) $O(h^2)$
D) O(log h)

4)**Which step determines Prim's improved time complexity with a Fibonacci heap?**
A) Extracting the minimum edge
B) Decreasing key operations
C) Union-Find operations
D) Sorting edges

5)**Adding an edge to a graph with a unique MST. When does the MST weight decrease?**
A) Always
B) If the new edge is lighter than all in the MST
C) If the new edge creates a cycle with a heavier edge
D) Never

6)**In a disconnected undirected graph with 10 nodes and 3 components, how many times does BFS restart?**
A) 1
B) 3
C) 10
D) 7

7)**A graph has all edges with weight 5. How many distinct MSTs exist if the graph is complete with 4 nodes?**
A) 1
B) 4
C) 16
D) 24

8)**Bidirectional BFS is most effective when:**
A) The graph is unweighted
B) The goal is to find all connected components
C) The target node is known, and the graph branches exponentially
D) The graph is a tree

9)**A: [B,C] B: [D] C: [E] D: [C] E: [], What is the finishing order (from first finished to last) if you run a recursive DFS starting at A and visiting neighbors in alphabetical order?**
A. A, B, D, C, E
B. E, C, D, B, A
C. D, B, E, C, A
D. B, D, E, C, A

10)**On the same graph, which of the following is a valid topological sort?**
A. A, B, D, C, E
B. A, C, E, B, D
C. B, A, D, C, E
D. D, B, A, C, E

Q2. You are given an m x n grid where each cell contains one of the following characters: 'U' (up), 'D' (down), 'L' (left), or 'R' (right). Each cell (i, j) has a directed edge to another cell based on its direction. For example, a cell with 'U' points to (i-1, j). If the edge leads outside the grid, the cell has no outgoing edge.

Return the number of distinct cycles in the grid.

grid = [

```
  ["R", "D"],
  ["L", "U"]
], (0,0) → (0,1) → (1,1) → (1,0) → (0,0) → cycle = 1
```

National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Design and Analysis of Algorithms	Course Code:	
Program:	BS(SE)	Semester:	6
Duration:		Total Marks:	20
Paper Date:	14 May, 2025	Weight	
Section:	SE-B	Page(s):	
Exam Type:	Quiz5-version1		

Student Name: **Roll No #:** **Section:**

Instructions: **Solve Question 2 on back side and Write Pseudo Code only**

1)**Edges: X→Y(6), X→Z(2), Z→Y(–3), Y→W(4), Z→W(7)**
Q: What's the true shortest distance X→W?
A) 3
B) 6
C) 7
D) 9

2)**In a dense graph (E ≈ V^2), which implementation of Dijkstra (array vs. binary-heap) asymptotically approaches the same running time?**
A) Array & heap both O(V^2)
B) Array O(V^2), heap O(E log V)
C) Array O(V log V), heap O(E log V)
D) Array O(E), heap O(V^2)

3)**Edges: P→Q(3), Q→R(–4), R→S(2), S→Q(1), P→S(10)**
Q: Running Bellman–Ford from P, on which pass do you first detect a negative-cycle relaxation?
A) 1st
B) 2nd
C) 3rd
D) never (no negative cycle)

4)**You're modeling flights with possible layovers, some of which offer time credits (negative layover time).Which approach finds the fastest itinerary and detects impossible loops?**
A) Pure Dijkstra
B) Bellman–Ford + cycle check
C) A* with zero heuristic
D) Bidirectional BFS

5)**In which graph-data scenario is Johnson's algorithm (Bellman–Ford + Dijkstra) strictly preferable?**
A) Single-source only
B) All-pairs on a sparse graph with negative edges
C) Dense graph with positives only
D) Dynamic streaming edges

6)**M→N(2), M→O(5), N→O(–3), O→P(4), N→P(10)**
What's the sequence of relaxations Bellman–Ford will perform on pass 2 to settle M→P?
A) M→N, N→O, O→P
B) M→O, O→P, N→P
C) N→O, O→P, N→P
D) M→N, N→P, O→P

7)**You have a graph where all edges are non-negative, but one edge weight is vastly larger than the others. Which behavior may degrade Dijkstra's performance in practice?**
A) Too many heap operations on small keys
B) Numeric overflow on large weights
C) Repeated extraction of a far-away node
D) Starvation of nodes with medium weights

8)**Which of these modifications can make Dijkstra correctly handle one negative edge without redesigning it entirely?**
A)Run Bellman–Ford only on that edge
B) Temporarily increase all weights by a constant
C) Reverse the graph and rerun
D) None—any negative edge breaks its greedy guarantee

9)**Road network augmented with live traffic delays (only increases, never negative). Which dynamic approach efficiently updates shortest paths without full recompute?**
A) Re-run Dijkstra from scratch
B) Delta-stepping variations of Dijkstra
C) Bellman–Ford incremental
D) Floyd–Warshall precompute

10)**How many full-edge-relaxation passes does Bellman–Ford make to compute shortest paths (before the cycle-check pass)?**
A) V
B) V–1
C) E
D) V+1

Q2. You are given a directed graph G = (V, E) with non-negative edge weights. Each edge represents a road between two cities, and the weight corresponds to the toll cost for that road. You are also given a source city s ∈ V. **Design an algorithm to compute the minimum total toll cost from the source city s to every other city in the graph, you can use a toll waiver on at most one edge in each path(algorithmically decide when to use the waiver). The toll waiver allows you to travel on one road without paying its toll, reducing that edge's cost to zero. This strategic use of the waiver can significantly impact the total cost of the journey.**
Cities: 1, 2, 3, 4
Roads:[(1, 2) with a toll cost of 10, (2, 3) with a toll cost of 10 ,(1, 3) with a toll cost of 50 ,(3, 4) with a toll cost of 10]
Path: 1 → 2 → 3 → 4 <u>Without using the toll waiver</u>:{1 to 2: 10} {2 to 3: 10} {3 to 4: 10} Total Cost: 10 + 10 + 10 = 30
<u>Using the toll waiver on edge (3, 4)</u>: {1 to 2: 10} {2 to 3: 10} {3 to 4: 0 (toll waived)} Total Cost: 10 + 10 + 0 = 20

National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Design and Analysis of Algorithms	Course Code:	
Program:	BS(SE)	Semester:	6
Duration:		Total Marks:	20
Paper Date:	14 May, 2025	Weight	
Section:	SE-B	Page(s):	
Exam Type:	Quiz5-version2		

Student Name: **Roll No #:** **Section:**

Instructions: **Solve Question 2 on back side and Write Pseudo Code only**

1)**Edges: A→B(4), A→C(2), B→D(7), C→D(3), D→E(1) What's the shortest distance A→E, and via which intermediate?**
A) 7 via A–B–D–E
B) 6 via A–C–D–E
C) 8 via A–B–D–E
D) 9 via A–C–D–E

2)**Edges: S→P(5), S→Q(2), P→R(−3), Q→R(4), R→T(3) Bellman–Ford settles S→T on which pass?**
A) 1st
B) 2nd
C) 3rd
D) never

3)**Edges: X→Y(3), Y→Z(5), X→Z(10), Z→W(−4), Y→W(2) After running Dijkstra, what's the tentative X→W before relaxing Z→W?**
A) 7
B) 8
C) 9
D) 10

4)**Dijkstra with a Fibonacci heap runs in O(E + V log V). For a sparse graph (E≈V), this is equivalent to which complexity?**
A) O(V log V)
B) $O(V^2)$
C) O(E log V)
D) $O(E^2)$

5)**Why does Bellman–Ford use exactly V–1 full edge-relaxation passes to compute shortest paths?**
A) To detect cycles
B) To propagate distances across the longest simple path of length V–1
C) Because of priority-queue limitations
D) Arbitrary but proven bound

6)**Which SSSP algorithm is best for a directed acyclic graph (DAG)?**
A) Dijkstra
B) Bellman–Ford
C) Topological-order relaxation
D) Floyd–Warshall

7)**A ride-sharing app models trips as graph edges, with surge pricing sometimes negative (discount).Which algorithm finds the cheapest route and flags impossible discount loops?**
A) Dijkstra
B) Bellman–Ford
C) A* with zero heuristic
D) Yen's k-shortest paths

8)**Live traffic updates only increase travel-time weights on edges. What dynamic technique updates shortest paths fastest without full recompute?**
A) Re-run Dijkstra from scratch
B) Delta-stepping incremental Dijkstra
C) Frequent Bellman–Ford passes
D) Precompute all-pairs and lookup

9)**Johnson's algorithm combines Bellman–Ford with Dijkstra to compute all-pairs shortest paths on sparse graphs with negative edges. What's its time complexity?**
A) O(VE + V E log V)
B) O(V E + V E log V)
C) O(VE + V E log V)
D) O(V E + E log V)

10)**You have one negative-weight edge but no cycles. Which single change lets you still use Dijkstra correctly?**
A) Add a large constant to all edges
B) Reverse edge directions
C) Preprocess that one edge with Bellman–Ford, then run Dijkstra
D) No change—Dijkstra fails with any negative edge

Q2.You are navigating a **directed weighted graph** representing a city map, where edges indicate roads and weights represent travel times in minutes. You start from a source node S.You are allowed to use a **one-time speed boost**, which lets you traverse **one edge** at **half its travel time** (rounded down if needed). The boost can only be used **once**, and only in the forward direction. Return an array of minimum travel times from the source node s to every other node, using the speed boost on at most one edge in each path. **Graph : [1 → 2 (8) 2 → 3 (4) 1 → 3 (20)]**

Source: 1 Path 1 → 2 → 3 = 8 + 4 = 12

Use boost on 1 → 2: 4 + 4 = 8(best) Use boost on 2 → 3: 8 + 2 = 10 Use boost on 1 → 3: 10

National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Design and Analysis of Algorithms	Course Code:	
Program:	BS(SE)	Semester:	6
Duration:		Total Marks:	20
Paper Date:	14 May, 2025	Weight	
Section:	SE-B	Page(s):	
Exam Type:	Quiz4-version1		

Student Name: **Roll No #:** **Section:**

Instructions: **Solve Question 2 on back side and Write Pseudo Code only**

1)**A: [B,C] B: [D] C: [E] D: [C] E: [], What is the finishing order (from first finished to last) if you run a recursive DFS starting at A and visiting neighbors in alphabetical order?**
A. A, B, D, C, E
B. E, C, D, B, A
C. D, B, E, C, A
D. B, D, E, C, A

2)**On the same graph, which of the following is a valid topological sort?**
A. A, B, D, C, E
B. A, C, E, B, D
C. B, A, D, C, E
D. D, B, A, C, E

3)**You want to preprocess a static weighted graph for many MST queries after deleting one edge each time. Which gives fastest per-query time?**
A. Run Kruskal from scratch each time: O(E log V)
B. Precompute MST and maintain dynamic tree structure: O(log V) per delete
C. Maintain adjacency matrix and recompute BFS: $O(V^2)$
D. Use Prim's with binary heap each time: O(E log V)

4)**You run BFS and DFS starting at node 1 on the same undirected graph. Which can produce the same spanning tree?**
A. Always
B. Never
C. Sometimes, if the graph is a tree
D. Only if the graph is complete

5)**A graph's edge weights are all distinct. Which yields a unique MST?**
A. Kruskal's
B. Prim's
C. Borůvka's
D. All of the above

6)**Which statement is false for a BFS tree rooted at s in an unweighted graph?**
A. It's a spanning tree of s's reachable vertices
B. The level of each vertex equals its distance from s
C. Every edge in the original graph connects vertices whose levels differ by at most 1
D. It minimizes the sum of distances in the tree

7)**You need to find the number of connected components in an undirected graph given as an edge list. Which is the fastest approach?**
A. Repeated BFS from unvisited nodes
B. Repeated DFS from unvisited nodes
C. Union-find on all edges, then count roots
D. Construct MST and count trees

8)**After Kruskal's, each disconnected component yields a:**
A) Hamiltonian cycle
B) Minimum spanning forest
C) DFS tree
D) Shortest-path tree

9)**To update an MST after deleting one edge, best worst-case per-query time uses:**
A) Kruskal from scratch O(E log V)
B) Dynamic-tree structure O(log V)
C) Re-run Prim's O(E log V)
D) Adjacency-matrix DFS $O(V^2)$

10)**Given E = $\Theta(V^2)$, Prim's with binary heap runs in:**
A) $O(V^2)$
B) O(E + V log V)
C) O(E log V)
D) O(V + E)

Q2. You are given an m x n grid where each cell contains one of the following characters: 'U' (up), 'D' (down), 'L' (left), or 'R' (right). Each cell (i, j) has a directed edge to another cell based on its direction. For example, a cell with 'U' points to (i-1, j). If the edge leads outside the grid, the cell has no outgoing edge.

Cells are part of the same connected component if you can reach them from one another by following directions (directed path).

Return the number of connected components in the grid. grid = [
["R", "L"],
["R", "L"]

], CC = 2 , → (0,0) “R” → (0,1) → (0,1) “L” → (0,0)

Section = BSE-8C

...ity of Computer and Emerging Sciences, Lahore Campus

Course:	Design and Analysis of Algorithms	Course Code:	
Program:	BS (Software Engineering)	Semester:	Spring 2025
Duration:	30 Minutes	Total Marks:	
Paper Date:		Weight	
Section:	BSE-8C	Page(s):	02
		Roll. No.	22L-7771
Exam:	Quiz 5	Name:	Hasan Yahya

...struction/Notes: Honesty always gives fruit and Dishonesty is always harmful.

1. What is the primary purpose of Breadth-First Search (BFS)?
   - A) To find the shortest path in a weighted graph.
   - B) To explore all vertices at the present depth before moving to the next level.
   - C) To sort vertices in topological order.
   - D) To find cycles in a graph.

2. In BFS, how are vertices typically colored to track their state?
   - A) Red, Green, Blue
   - B) White, Gray, Black
   - C) Unvisited, Processing, Processed
   - D) Undiscovered, Discovered, Finished

3. What is the time complexity of BFS on a graph with V vertices and E edges?
   - A) O(V^2)
   - B) O(V+E)
   - C) O(ElogV)
   - D) O(VlogE)

4. Which of the following is a property of a Minimum Spanning Tree (MST)?
   - A) It contains all vertices of the graph and has the maximum total edge weight.
   - B) It is unique for every graph.
   - C) It connects all vertices with the minimum total edge weight and no cycles.
   - D) It is only applicable to directed graphs.

5. Which algorithm for finding an MST grows the tree by adding the cheapest edge connecting the tree to a new vertex?
   - A) Kruskal's algorithm
   - B) Prim's algorithm
   - C) Dijkstra's algorithm
   - D) Bellman-Ford algorithm

6. In Kruskal's algorithm, how are edges processed?
   - A) In arbitrary order.
   - B) In order of increasing weight.
   - C) In order of decreasing weight.
   - D) Based on vertex degrees.

7. What is the key data structure used in Kruskal's algorithm to efficiently check for cycles?
   - A) Stack
   - B) Queue
   - C) Disjoint-set (Union-Find)
   - D) Priority queue

8. Which theorem justifies the "safe edge" strategy used in both Prim's and Kruskal's algorithms?

CQs no.	1	2	3	4	5	6	7	8
Ans	A	B	B	C	B	B	C	C

- A) Master Theorem
- B) Cut Property Theorem
- C) Pigeonhole Principle
- D) Central Limit Theorem

Q2: Explain how a vertex u of a directed graph can end up in a depth-first tree contain ing only u, even though u has both incoming and outgoing edges in G.

→ One of the ways this can happen is if the node connects to itself ie is self looping

it has both incoming or outgoing edges, but it will always only work after a restart and after it a restart will happen. So, it will be in its own tree in DFS forest.

→ otherways is if graph is like this,

only one bridge

This graph has 2 bridges. The seperate D, so it to allow be in its own tress

→

f. works

not unique

National University of Computer and Emerging Sciences, Lahore Ca

Section: B

Course:	Design and Analysis of Algorithms	Course Code:	
Program:	BS (Software Engineering)	Semester:	
Duration:	30 Minutes	Total Marks:	
Paper Date:		Weight	
Section:	BSE-6C	Page(s):	
Exam:	Quiz 6	Roll. No.	02
		Name:	

6

Instruction/Notes: Honesty always gives fruit and Dishonesty is always harmful.

1. What is the primary purpose of the Bellman-Ford algorithm?

- A) To find the shortest paths in a graph with nonnegative edge weights.
- B) To detect negative-weight cycles reachable from the source.
- C) To compute the longest path in a directed acyclic graph (DAG).
- D) To sort vertices topologically.

2. What is the time complexity of the Bellman-Ford algorithm for a graph with V vertices and E edges?

- A) $O(V \log V)$
- B) $O(VE)$
- C) $O(V+E)$
- D) $O(E \log V)$

3. In Dijkstra's algorithm, which data structure is most efficient for sparse graphs?

- A) Array
- B) Binary min-heap
- C) Fibonacci heap
- D) Queue

4. Which algorithm is best suited for finding shortest paths in a Directed Acyclic Graph (DAG)?

- A) Dijkstra's algorithm
- B) Bellman-Ford algorithm
- C) Topological sort followed by edge relaxation
- D) Breadth-First Search (BFS)

4

5. What invariant does Dijkstra's algorithm maintain?

- A) All vertices in the priority queue have correct shortest-path weights.
- B) The set S contains vertices with finalized shortest-path weights.
- C) The predecessor subgraph is always a BFS tree.
- D) Edge weights are relaxed in arbitrary order.

6. Why does Dijkstra's algorithm fail with negative-weight edges?

- A) It assumes edge weights are integers.
- B) It may incorrectly assume a path is shortest before processing negative edges.
- C) It cannot detect cycles.
- D) It requires a priority queue.

7. What property ensures correctness in the DAG shortest-paths algorithm?

- A) All edge weights are positive.

No:	1	2	3	4	5	6	7	8
ns	A	B	B	A	A	B	B	A
	✗	✓	✗	✗	✗	✓	✓	✓

- B) No cycles allow processing vertices in topological order.
- C) The graph is undirected.
- D) The algorithm uses BFS.

---

8. How does the Bellman-Ford algorithm detect negative-weight cycles?

- A) By checking if any edge can be relaxed after V−1 iterations.
- B) By running Dijkstra's algorithm twice.
- C) By sorting edges by weight.
- D) By using a Fibonacci heap.

Q2: Suppose that we are given a weighted, directed graph G = (V, E) in which edges that leave the source vertex s may have negative weights, all other edge weights are nonnegative, and there are no negative-weight cycles. Argue that Dijkstra's algorithm correctly finds shortest paths from s in this graph.

It does not.

2

while dijkstras algorithm does find the shortest path, these paths will always account for negative edges/ weights. which means that the final cost will not be the same as it was positive

→ It is not possible to find non-negative weight path (shortest) at-all in any form if we use Dijkstras is any-