**National University of Computer & Emerging Sciences, Karachi Fall-2021 Department of Computer Science** 

**Final Exam**

**29 December 2021, 09:00 AM – 09:30 AM**

**Part (A)**

| **Course Code:** CS2009 | **Course Name:** Design and Analysis of Algorithm | |
| --- | --- | --- |
| **Instructor Name / Names:** Dr. Muhammad Atif Tahir, Dr. Fahad Sherwani, Dr. Farrukh Saleem, Waheed Ahmed, Waqas Sheikh, Sohail Afzal | | |
| **Student Roll No:** | | **Section:** |

Instructions:

• Must be submitted with in 30 minutes

• No extra sheets allowed. Must solve in given space

• You are allowed to submit paper before 30 minutes and start part(b)

**Time**: 30 minutes **Max Marks: 10**

**Question # 1 [0.5\*6 = 3 marks]**

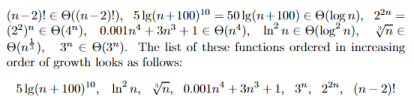
**Answer the following questions. You must explain in only 3-4 lines.**

(a) Describe Big theta.

c1g(n)<=f(n)<=c2g(n)

(b) List the following functions according to their order of growth from the lowest to the highest.

3, 3��.

(�� − 2)!, 5 log( �� + 100)10, 22��, 0.001 ��4 + 3��3 + 1, ln2 �� , √�� 

(c) Explain why the statement, "The running time of algorithm A is at least ��(��2)," is meaningless.

We just care about the upper bound and the lower bound of T(n)*T*(*n*).

The statement: T(n)*T*(*n*) is at least O(n^2)*O*(*n*2).

• Upper bound: Because "T(n)*T*(*n*) is at least O(n^2)*O*(*n*2)", there's no information about the upper bound of T(n)*T*(*n*).

• Lower bound: Assume f(n) = O(n^2)*f*(*n*)=*O*(*n*2), then the statement: T(n) \ge f(n)*T*(*n*)≥*f*(*n*), but f(n)*f*(*n*) could be any fuction that is "smaller" than n^2*n*2. For example, constant, n*n*, etc, so there's no conclusion about the lower bound of T(n)*T*(*n*), too.

Therefore, the statement, "The running time of algorithm A*A* is at least O(n^2)*O*(*n*2)," is meaningless.

(d) Define recurrence relations and enlist methods to solve them.

An equation or inequality that describes a function in terms of its value on smaller inputs. 1. Recursion tree method 2 Iteration method Substitution method 4 Master method

(e) In which conditions dynamic programing does not work. Give suitable example. Basic property of problems which can be solved using DP are.

1. Overlapping Sub-problems.

2. 2.Optimal Sub-structure

If any problem doesn't have either of above property then problem can't be solved using DP. for example shortest path can be solved using DP but longest path can't solved, because longest path doesn't hold optimal sub-structure property

(f) Suppose there is a maximization problem, where the approximate solution has the cost of 25

And optimal solution has the cost of 30. Find the approximation ratio.

30/25 = 1.2 approximation ratio

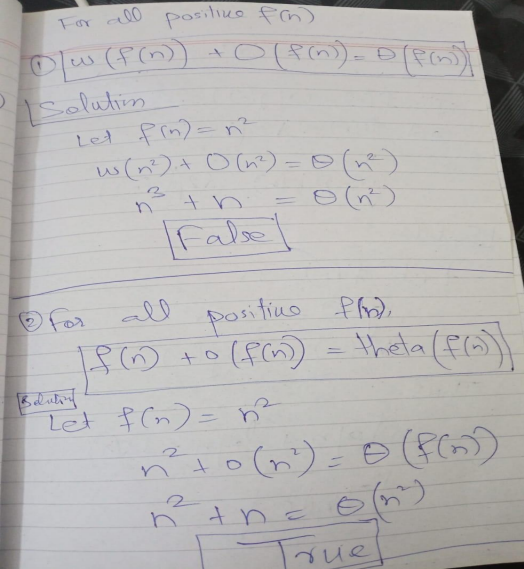
**Question # 2 [0.5\*8 = 4 marks]**

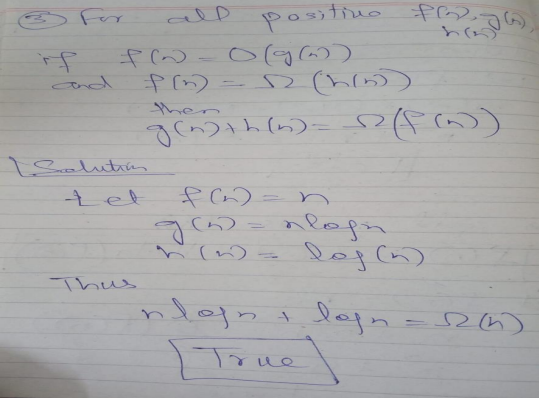
Write the complexity and the corresponding design strategy (Divide and Conquer/ Dynamic Programming / Greedy) of the given algorithms

| ALGORITHMS | Worst Case | Write below whether the  algorithm belongs to Divide and Conquer/ Dynamic Programming / Greedy / None of them. If an algorithm can be solved with several design techniques, write anyone of them with the  corresponding complexity. |
| --- | --- | --- |
| Quick Sort | ��(��^2) | Divide and Conquer |
| Radix Sort | O((n+k)\*d) | None of them |
| Max-Heapify operation | ��(��������) | None of them |
| Add Vertex in Adjacency Metric | ��(��^2) | Greedy/ None of them |
| Rod-Cutting | ��(��^2) / O  (2^n) | Dynamic Programing/ None of them |
| Dijkstra’s (Array DS) | V^2 | Greedy |
| Prims | Elog V | Greedy |
| Maximum Sub-array Sum | O(n)/ O(nlogn) /O (n^2) | Dynamic/ divide and Conquer/ None of them. |

**Question # 3 [1\*3 = 3 marks]**

For each of the following questions indicate whether it is true or false and justify using some example assuming a function.





**National University of Computer & Emerging Sciences, Karachi Fall-2021 Department of Computer Science** 

**Final Exam**

**29 December 2021, 09:00 AM – 12:00 AM**

**Part (B)**

| **Course Code:** CS2009 | **Course Name:** Design and Analysis of Algorithm | |
| --- | --- | --- |
| **Instructor Name / Names:** Dr. Muhammad Atif Tahir, Dr. Fahad Sherwani, Dr. Farrukh Saleem, Waheed Ahmed, Waqas Sheikh, Sohail Afzal | | |
| **Student Roll No:** | | **Section:** |

Instructions:

• Return the question paper.

• Read each question completely before answering it. There are **9 questions** on **4 pages**. • In case of any ambiguity, you may make assumption. But your assumption should not contradict any statement in the question paper.

•

**Time**: 180 minutes **Max Marks: 40**

**Question # 4 [6 marks]** a) What is meant by P and NP Problems?

• P problems

– (The original definition) Problems that can be solved by **deterministic Turing machine** in polynomial-time.

– (A equivalent definition) Problems that are solvable in polynomial time.

• NP problems

– (The original definition) Problems that can be solved by **non-deterministic Turing machine** in polynomial-time.

– (A equivalent definition) Problems that are **verifiable** in polynomial time.

• Given a solution, there is a polynomial-time algorithm to tell if this solution is correct.

b) Let X be a problem that belongs to the class NP. Then explain why the following are incorrect or correct statements?

A. Explain why it is incorrect) There is no polynomial time algorithm for X B. (Explain why it is incorrect) If X can be solved deterministically in polynomial time then P = NP

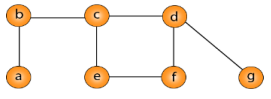
C. (Explain why it is correct) Since If X is NP-hard, then it is NP-complete D. (Explain why it is incorrect) X may be undecidable

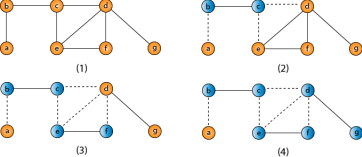
• Sol: (A) is incorrect because set NP includes both P( Polynomial time solvable) and NP-Complete . (B) is incorrect because X may belong to P (same reason as (A)) (C) is correct because NP Complete set is intersection of NP and NP-Hard sets. (D) is incorrect because all NP problems are decidable in finite set of operations.

c) Does P ! = NP mean that no problem exists which can be solved and checked in polynomial time?

No. P != NP means that P is a proper subset of NP. That means, there are problems that are in NP but not in P.

d) Using 2-approximation greedy method studied during lectures, find the size of the vertex cover of the following graph:

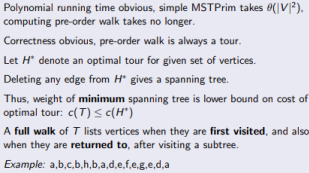


**Question # 5**

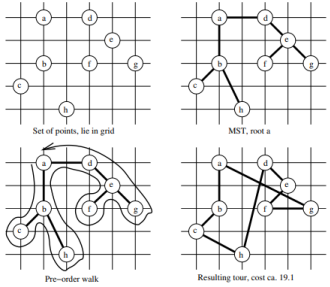
**[4 marks]**

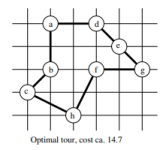
Prove that Travelling Salesman approximation algorithm is a 2-approximation algorithm**.** Give arguments and example as well.

Solution:







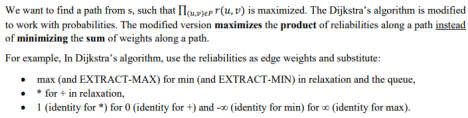


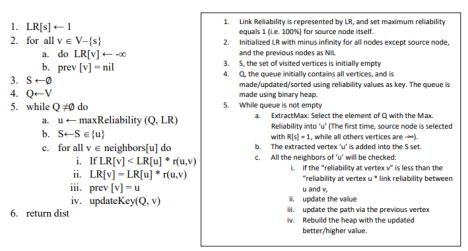
**Question # 6 [4 marks]**

We are given a directed graph G = (V, E) on which each edge (u,v) ϵ E has an associated value r (u,v), which is a real number in the range 0 ≤ r(u,v) ≤ 1 that represents the reliability of a communication channel from vertex u to vertex v. We interpret r(u,v) as the probability that the channel from u to v will not fail, and we assume that these probabilities are independent. Give an efficient algorithm to find the most reliable path between two given vertices.

*(Note: Reliability means higher value)*

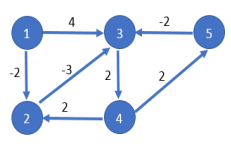
Solution:



**Question # 7 [3 + 1 + 1 = 5 marks]**

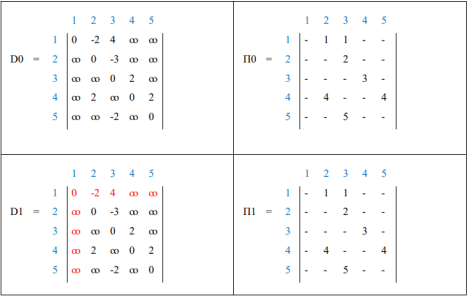
(a) Use Floyd Warshall algorithm to find the shortest path from every vertex to every other vertex for the graph given below. The result must contain two matrices/tables, one matrix (D) shows the shortest cost from each vertex to all other vertices, and the second matrix (Π) should show the previous vertex use to reach the destination vertex.

(b) Mark the steps on the resultant Π matrix/table to show, how it will be used, to trace the route from vertex 5 to vertex 2.

(c) From the resultant matrices, identify all those pairs of vertex for which there is no path.

Solution:

Use Floyd Warshall algorithm to find the shortest path from every vertex to every other vertex for the graph given below. The result must contain two matrices/tables, one matrix (D) shows the shortest cost from each vertex to all other vertices, and the second matrix (Π) should show the previous vertex use to reach the destination vertex.





(b) For the Π matrix/table given below, the empty cells in-fact represents NIL (/) value. Which means that a path does not exists. Mark the steps on the resultant Π matrix/table to show, how it will be used, to trace the route from vertex 5 to vertex 2.



In the resultant Π table, to reach from vertex 5 to vertex 2, we look at the cell (5,2), which contain the previous vertex 4 to reach vertex 2. Now, to reach vertex 4, we look at the cell (5,4), which contain previous vertex 3 to reach vertex 4 (as shown by step 2). Finally, to reach vertex 3, we look at cell (5,3), which contain previous vertex 5 (self), that is directly connected (as shown by step 3).

(c) From the resultant matrices, identify all those pairs of vertex for which there is no path. Solution:

2 -> 1, 3 -> 1, 4 -> 1, 5 -> 1, and all self-nodes

**Question # 8 [2+3 = 5 marks]**

Let array A= *A*[1], *A*[2], …, *A*[*n*] stores the number of cars produced by a company in the past *n* years,. The company wants to find if there is a period of consecutive years (from i to j), such that the total number of cars produced in this period is exactly equal to *M*. if there exist *i* and *j*,

*j*

1 ≤ *i* ≤ *j* ≤ *n*, such that ∑

*A*[*k*]= *M*.

*k i*

=

Example : Input : A [2, 4, 6, 3, 8, 4, 1, 10], M= 15

a) Design brute force algorithm for this problem b) Design O(*n*) time greedy algorithm to solve this problem.

Solution:



**Question # 9 [2+3 = 5 marks]**

The dots in below *Figure(a)* represent a collection of towns, and the edges between the towns in *Figure(b)*, shows that the two towns u and v with the edge (u,v) are at-most 30 miles apart. The town committee is deciding where to open the schools.

There are only two constraints:

a) Each school should be in a town.

b) No student should have to travel more than 30 miles to reach one of the school.

With the above given constraints, design approximation algorithm to determine the minimum number of schools needed?



Solution:

This is a typical *set cover* problem. For each town x, let Sx be the set of towns within 30 miles of it. A school at x will essentially cover. these other towns. The question is then, how many sets Sx must be picked in order to cover all the towns in the county?

SET COVER

*Input:* A set of elements B; sets S1;… Sm \_ B

Output: Solution set C

U = B

C = Φ

While (U is not empty)

������������ set Si with the largest number of uncovered elements

U =U - ����

C = C U {S}

End while

Return C

Mark all the input sets as uncovered

Make an empty solution set U

Repeat until all elements of B are covered:

Pick the set Si with the largest number of uncovered elements.

Place the Si in U

Mark all sets connected to Si

Return U

**Question # 10 [ 3 marks]**

Construct failure function table (that we build up in KMP string matching algorithm) for given pattern :



Solution:

The failure function table with computational steps:







**Question # 11 [1+3 =4 marks]** ⃗⃗⃗⃗⃗⃗⃗⃗⃗⃗⃗ in below figure, design a brute force algorithm to

a. Given the line segment (��, ��)

determine whether this line will be the part of convex hull. Hint: use counterclockwise turn technique.

Solution:







Dry run

⃗⃗⃗⃗⃗⃗⃗⃗⃗⃗⃗ to all other points and check whether it is positive or

Compute ccw of line segment (��, ��)

not

1. a = [3, 3] , b= [-2, 2] , c= [-2, 0]

( -2 - 3 ) \* ( 0 - 3 ) - ( -2 - 3 ) \* ( 2 - 3 )

Ans= 10

its counter clockwise turn

2. a = [3, 3] , b= [-2, 2] , c= [0, 0]

( -2 - 3 ) \* ( 0 - 3 ) - ( 0 - 3 ) \* ( 2 - 3 )

Ans= 12

its counter clockwise turn

3. a = [3, 3] , b= [-2, 2] , c= [1, 1]

( -2 - 3 ) \* ( 1 - 3 ) - ( 1 - 3 ) \* ( 2 - 3 )

Ans= 8

its counter closckwise turn

4. a = [3, 3] , b= [-2, 2] , c= [5, 2]

( -2 - 3 ) \* ( 2 - 3 ) - ( 5 - 3 ) \* ( 2 - 3 )

Ans= 7

its counter closckwise turn

**Question # 12 [2+2 =4 marks]**

Compute time complexity :

(a)

**This is O(**��������(��)**)**

Although there is only one loop but at each iteration of while loop , “i” is incremented and then added to previous value of “s” which makes it **O(**��������(��)**)**

(b)

**This is *O*(n*)***

Only first for loop will run (n/3) times so ***O(n)***

For each iteration of first for loop, second for loop will run for one time only (as j will be equal to n in second for loop in first iteration and j<n condition will become false) and second for loop will terminate after its every first iteration.

It will also not go in the inner most while loop as while condition (j<=1) will never be true .