National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	**Design and Analysis of Algorithms**	**Course Code:**	**CS2009**
	Degree Program:	**BSCS**	**Semester:**	**Fall 2021**
	Exam Duration:	**180 Minutes**	**Total Marks:**	**58**
	Paper Date:	**14 Jan 2022**	**Weight**	**50**
	Section:	**ALL**	**Page(s):**	**9**
	Exam Type:	**Final Exam**		

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

Instruction/Notes: Attempt the examination on the question paper and write concise answers. You can use extra sheet for rough work. Do not attach extra sheets used for rough with the question paper. Do not fill the table titled Questions/Marks.

Question	1	2-5	6-7	8-11	12	13	14	Total
Marks	/5	/9	/10	/10	/7	/7	/10	/58

Q1) Solve the following recurrence and find out asymptotic time complexity. Show complete working. [5 Marks]

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

Q2) What is the most efficient sorting algorithm for sorting 1 million numbers. The data is in the range -100 to 100. [2 Marks]

a) Insertion sort
b) Quick sort
c) Merge sort
d) Count sort
e) Heap sort

Q3) Suppose we have a O(n) time algorithm that finds median of an unsorted array. Now consider a QuickSort implementation where we first find median using the above algorithm, then use median as pivot. What will be the worst case time complexity of this modified QuickSort? [2 Marks]

Q4) When does the worst case of Quick sort occur? [2 Marks]

Q5) Describe an algorithm that, given n integers in the range 0 to k, preprocesses its input and then answers any query about how many of the n integers fall into a range [a . . b] in O(1) time. Your algorithm should use $\theta(n + k)$ preprocessing time. [3 Marks]

Q6) A sub-sequence of a given sequence is just the given sequence with some elements (possibly none or all) left out. We are given two sequences X[m] =$x_0,x_1,...x_{m-1}$ and Y[n] = $y_0,y_1,...,y_{n-1}$of lengths m and n respectively, with indexes of X and Y starting from 0. We wish to find the length of the longest common sub-sequence(LCS) of X[m] and Y[n] as LCS(m,n), where an incomplete recursive definition for the function LCS(i,j) to compute the length of The LCS of X[m] and Y[n] is given below: [5 Marks]

LCS(i,j) = 0, if either i=0 or j=0

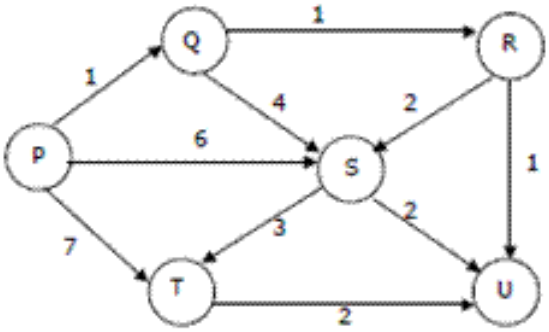
= expr1, if i,j > 0 and x_{i-1} **=** y_{j-1}

= expr2, if i,j > 0 and x_{i-1} **!=** y_{j-1}

What is value of expr1 and expr2 ?

Q7) This question is about maximum subarray sum problem. We are given an integer array A[0..n) and we have to find the subarray A[i..j) such that the sum of the integers A[i] + A[i + 1] + · · · + A[j − 1] is maximal. If M[j] holds the maximum sum for any segment A[i..j) ending in j − 1 how can we efficiently compute the maximum sum for any segment A[i…..j + 1) ending in j? In other words, write the expression for defining the value of M[j+1].
[Hint: you do not need a loop or recursive function.] [5 Marks]

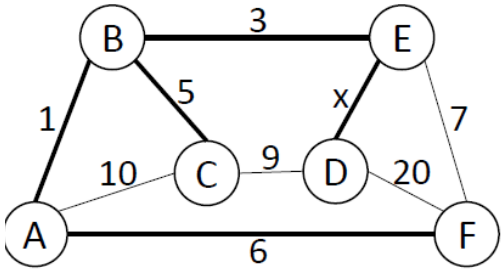
Q8) Suppose we run Dijkstra's single source shortest-path algorithm on the following edge weighted directed graph with vertex P as the source. In what order do the nodes get included into the set of vertices for which the shortest path distances are finalized? [2 Marks]



a) P Q T S R U
b) P Q T S U R
c) P Q R S U T
d) P Q R U S T
e) P S Q R U T

Q9) For the following graph the bold edges form a Minimum Spanning Tree. What can you tell about the range of values for x? [2 Marks]

a. Less than 7, greater than 10
b. Less or equal to 9
c. Less or equal to 3
d. Less or equal to 7



Q10) If a graph G has some negative edge weights, Dijkstra's algorithm does not guarantee to compute shortest paths. However, would finding the minimum weight (most negative weight) w and adding the absolute value to every edge's weight solve this problem? In other words, if we normalize all edge weights by adding a constant absolute value to each edge such that the smallest edge weight in new graph is 0 then will Dijkstra'a algorithm correctly compute shortest paths on this new graph? Justify your answer by giving example. [3 Marks]

Q11) We have a weighted undirected graph G and its minimum spanning tree T. Suppose we create a new graph H by adding a constant weight w to the weight of all the edges in G. Do the edges of T also represent the minimum spanning tree of H? Justify your answer or provide a counter-example. [3 Marks]

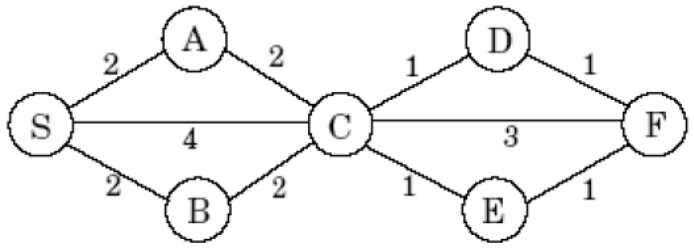
Q12) Given an array A of integers of size *n*, and a number b, design an algorithm that finds two integers x and y from the array such that x + y = b. Your algorithm should run in O(n log n) time. For partial credit, you can design an algorithm that runs in O(n^2) time. [7 Marks]

Q13) You have an unsorted array, A, of unique numbers. You know its median number in advance. Now you wish to arrange the elements of A in such a way that the numbers $A[i]$ and $A[i+1]$ are successively in the increasing and decreasing order. That is, $A[1] < A[2]$, $A[2] > A[3]$, $A[3] < A[4]$… and so on. A friend of yours claims that this is possible in $O(n)$. Is her claim True / False. Justify your answer. [7 Marks]

Q14) In cases where there are several different shortest paths between two nodes (and edges have varying lengths), the most convenient of these paths is often *the one with fewest edges*. For instance, if nodes represent cities and edge lengths represent costs of flying between cities, there might be many ways to get from city s to city t which all have the same cost. The most convenient of these alternatives is the one which involves the fewest stopovers. Accordingly, for a specific starting node s, define

best[u] = minimum number of edges in a shortest path from s to u

In the example below the best values for nodes S, A, B, C, D, E, F are 0,1,1,1,2,2,3 respectively.



Give an efficient algorithm for the following problem.
Input: Directed Graph G = (V, E); positive edge lengths l_e; starting node $s \in V$
Output: The values of best[u] should be set for all nodes $u \in V$ [10 Marks]

a) Briefly explain your algorithm in English.

b) Write pseudo code of your algorithm

c) Analyze its worst case time complexity.