National University of Computer and Emerging Sciences

FAST School of Computing **Fall-2022** **Islamabad Campus**

CS-2009: Design and Analysis of Algorithms (SE P,Q,R,S)

Thursday, 10th November, 2022

Course Instructor

Noor ul Ain, Bilal Khalid Dar

Serial No:

Sessional Exam-II

Total Time: 1 Hour
Total Marks: 70

_____
Signature of Invigilator

_____ _____

Student Name Roll No. Section Signature

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Attempt on question paper. Attempt all of them. Read the question carefully, understand the question, and then attempt it. In case of any ambiguity write down your assumption and solve the question.
2. No additional sheet will be provided for rough work. Use the back of the last page for rough work.
3. If you need more space write on the back side of the paper and clearly mark question and part number etc.
4. After asked to commence the exam, please verify that you have nine (09) different printed pages including this title page. There are a total of 6(Six) questions.
5. Calculator sharing is strictly prohibited.
6. Use permanent ink pens only. Any part done using soft pencil will not be marked and cannot be claimed for rechecking.

	Q-1	Q-2	Q-3	Q-4	Q-5	Q-6	Total
Marks Obtained							
Total Marks	**18**	**08**	**09**	**05**	**20**	**10**	**70**

Question 1 [18 Marks]

a. What value of q does PARTITION return when all elements in the array A [p . . . r] have the same value? Explain it briefly [3]

Partition always return q as r, where r is the size of the partition (i.e. passed to the partition algorithm). So the array gets divided by n-1 and 1 element that leads to the worst case synario.

b. HEAPSORT, as we studied it, puts its output in an array and sorts in ascending order. Explain how to modify HEAPSORT so that it creates a linked list in descending order. Your modifications should be as efficient as possible and not use any memory that is not necessary. Please note that you do not need to write algorithm and your explanation is enough. [4]

Use Min heap instead of Max heap
+explanation of memory usage

c. Give an array for which HEAPSORT will perform better than QUICKSORT. Explain why, using asymptotic running times. [3]

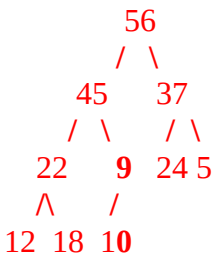
HEAPSORT runs in O(n lg n) worst case time.
QUICKSORT'S average case is $\Theta((n \lg n)$, but its worst case of $O(n^2)$ occurs when its input is already sorted, because it produces splits where one of the "conquer" calls is on an empty array each time.
So, (100, 99, 98, 97, …, 1) is one such array.

d. Given choice of four algorithms (insertion sort, randomized quick sort, bucket sort, counting sort), which is preferred and why for the given sets of data? [8]

Data	Sorting Algorithm (Reason)
{1,3,2,1,2,3,1,1}	Counting Sort (values are in a small range)
{1000,945,930,600,200,198,88}	Insertion Sort (Values are already sorted)/Randomized Quick Sort
{1,1000,3,10234,12345,65987}	Insertion Sort (The list is almost sorted (except one number)
{100, 10,500,250,302,156,404}	Bucket Sort (The data is uniformly distributed)/Randomized Quick Sort

Question 2 [08 Marks]

a. Is the array with values {56, 45, 37, 22, 9, 24, 5, 12, 18, 10} a max-heap? Show with the help of the binary heap data structures [2]



(No, this is not a max heap due to violation of max heap property at node no 5)

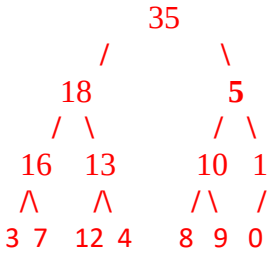
b. In worse case what will be the height of a Binary Heap Tree containing 32 elements? [1]

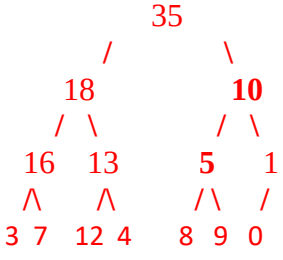
$Log_2(32)= 5$

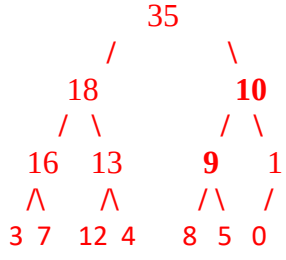
c. Illustrate the operation of MAX-HEAPIFY (A, 3) on the following array [5]

A = { 35, 18, 5, 16, 13, 10, 1, 3, 7, 12, 4, 8, 9, 0}

```
MAX-HEAPIFY(A, i)
1  l = LEFT(i)
2  r = RIGHT(i)
3  if l ≤ A.heap-size and A[l] > A[i]
4      largest = l
5  else largest = i
6  if r ≤ A.heap-size and A[r] > A[largest]
7      largest = r
8  if largest ≠ i
9      exchange A[i] with A[largest]
10     MAX-HEAPIFY(A, largest)
```







Question 3 [09 Marks]

a. Suppose that in a graph for all edges e ε E and w(u,v)=1 for (u,v) ε E. Can Dijkstra's algorithm be improved to find the shortest path? Explain it. What will be the asymptotic time complexity of the improved version? [3]

- Use a simple FIFO queue instead of a priority queue.

Breadth-first search

```
while Q ≠ ∅
  do u ← DEQUEUE(Q)
     for each v ∈ Adj[u]
        do if d[v] = ∞
              then d[v] ← d[u] + 1
                   ENQUEUE(Q, v)
```

Analysis: Time = $O(V + E)$.

b. When do we need to use Bellman-Ford algorithm instead of Dijkstra's algorithm for "Single Source Shortest Path" problem in a graph? Explain with the help of a graph example where Dijkstra's algorithm does not provide correct results and instead we need to use Bellman-Ford algorithm. [4]

Yes, in case of graph with negative weights, Bellman Ford algorithm is used instead of Dijkstra.(1)
Graph Example (2) with shortest path value for both Dijkstra and Bellman Ford (1)

c. Suppose that some of the weights in a connected graph G are negative. Will Prim's algorithm still work? What about Kruskal's algorithm? [2]

Yes, both algorithms will work for negative weights

Question 4 [05 Marks]

Considering KMP algorithm, please make a pi table for the following pattern,

P = A C A C A G A A A C A

```
COMPUTE-PREFIX-FUNCTION(P)
 1  m = P.length
 2  let π[1..m] be a new array
 3  π[1] = 0
 4  k = 0
 5  for q = 2 to m
 6      while k > 0 and P[k + 1] ≠ P[q]
 7          k = π[k]
 8      if P[k + 1] == P[q]
 9          k = k + 1
10      π[q] = k
11  return π
```

A	C	A	C	A	G	A	A	A	C	A
0	0	1	2	3	0	1	1	1	2	3

Question 5 [20 Marks]

Use finite automata string matching to show if the following pattern exist in the given text

T = A B C D A B C A C A C A C A C G A A

P = A C A C A G A

Please show the complete working (table for state machine and graphical representation of state machine)

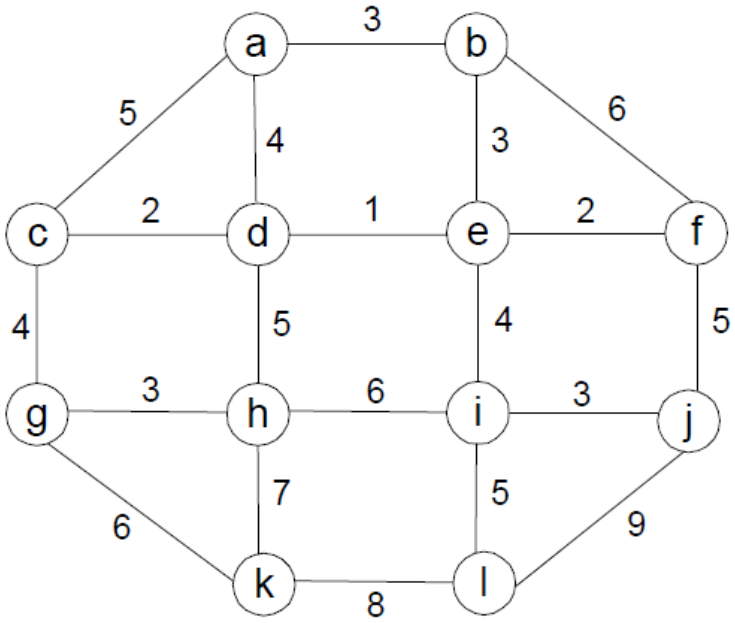
	A	C	G
A	**1**	**0**	**0**
C	**1**	**2**	**0**
A	**3**	**0**	**0**
C	**1**	**4**	**0**
A	**5**	**0**	**0**
G	**1**	**4**	**6**
A	**7**	**0**	**0**
	1	**2**	**0**

Students will make state machine

Pattern does not exist

Question 6 [10 Marks]

Apply Prim's algorithm to the following graph to find the minimum spanning tree



Pseudocode for Prim's Algorithm MST-Prim(G, r)

```
1 for each u ∈ V[G]
2     key[u] = ∞
3     π[u] = NIL
4 key[r] = 0
5 Q = V(G)
6 while Q is not empty
7         u= EXTRACT-MIN(Q)
8         for each v ∈ Adj[u]
9             if v ∈ Q and w(u, v) < key[v]
10                DECREASE-KEY(Q, key[v], w(u, v))
11                π[v] = u
12 return {(v, π[v]) : v ∈ V − {r}}
```

The minimum spanning tree found by the algorithm comprises the edges ab, be, ed, dc, ef, ei, ij, cg, gh, il, gk.

Total cost = 36

The END.