Assignment # 2

(CS-2009 Design and Analysis of Algorithms – Fall-2024)

Due Date and Time: 31 Oct 2024 Marks: 135

Submission Guideline:

- Your answers should be handwritten. You have to submit your assignment in both hard copy and soft copy.
- Mention your Name, Roll no and Section on front page.
- Submit your hard copy in A209F or A209E till 3pm (Thursday 31 Oct).
- You have to submit soft copy on GCR.
- Use camscanner application to compile your assignment in a pdf format in a single file. Name it as your name and roll no and submit on GCR in due time.
- No late submissions will be accepted.

Q1) **(4+4+4+4 = 16 Marks)**

a) Give at least four reasons as to why someone would use merge sort over quicksort ?

b) When will the Quicksort perform poorly (worst case) , Explain?

c) We have a system running insertion sort and we find that it's completing faster than expected. What could we conclude about the input to the sorting algorithm?

d) Explain which sorting algorithm you would use to sort the input array the fastest and why you chose this sorting algorithm. An array of n Comparable objects that is sorted except for k randomly located elements that are out of place (that is, the list without these k elements would be completely sorted).

Q2) **(2+2+4= 8 Marks)**

a) Does this array represent a Max-Heap with the root at index 1? (Ignore the data at index 0.)

index	0	1	2	3	4	5	6	7	8	9	10	11	12
Data	12	78	50	30	8	65	22	28	4	6	12	2	13

b) Consider 21, 20a, 20b, 12, 11, 8, 7 elements, show that whether the Heapsort is stable or not?

c) Consider the following algorithm for building a Heap of an input array A. Solve in step by step the time complexity of building a heap function below and show it takes O(n) time:

```
BUILD-HEAP(A)
 heapsize := size(A);
 for i := floor(heapsize/2) downto 1
 do HEAPIFY(A, i);
 end for
END
```

Q3) **(5+10 marks)**

Given a data structure S that allows the following four operations, all of which run in constant time O(1):

- S.addToFront(item) — Inserts an element at the front of the sequence.
- S.removeFromFront() — Removes and returns the element at the front of the sequence.
- S.addToBack(item) — Inserts an element at the end of the sequence.
- S.removeFromBack() — Removes and returns the element at the end of the sequence.

Where delete operations return the deleted element. Using only these operations, design algorithms that perform the following tasks efficiently. Your algorithms should rely solely on the above four operations, and each task should be completed with minimal overhead:

a) swapEnds(S): Swap the first and last elements in the sequence S using O(1) time.
b) shiftLeft(S, k): Move the first k elements of the sequence S to the end, preserving their order, in O(k) time. After execution, the k-th element should be last, and the element originally in position k+1 should be first.

Q4) (10+10 Marks)

A well-known organization, **PakFlora**, hosts an annual competition that evaluates home and community gardens from various regions of Pakistan. Each garden is assigned a score, represented by si, which is a positive integer reflecting its rating. Additionally, every garden is given a unique identifier ri, a positive integer serving as its registration number.

a) In an effort to promote inclusivity and reduce competitive pressure, **PakFlora** has decided to award identical trophies to the top k gardens. Given an unsorted list A containing garden pairs, and a positive integer *k* such that $k \leq |A|$, outline an algorithm (don't write code) that runs in $O(|A|+k\log|A|)$ time to return the registration numbers of the k highest-scoring gardens, resolving any ties arbitrarily.

b) **PakFlora** has decided to take a more objective approach and award a trophy to every garden with a score strictly greater than a reference score x. In a max-heap structure A that holds pairs of garden attributes, where each pair includes a numerical score and a corresponding unique registration number, describe a method (without providing code) that runs in $O(n_x)$ time to extract the registration numbers of all gardens that have scores exceeding a given threshold value x. Here, n_x represents the count of gardens that fulfill this scoring criterion.

Q5) (5+5)

1. Explain with examples the best and worst case scenario of the Naive String matching algorithm.
   Marks will only be given for properly writing each step with notations.
2. We have studied multiple string matching algorithms in class and have found naive to be an inefficient algorithm.
   - Can you suggest a solution to increase the efficiency of the naive algorithm?
   - Will it be feasible for all the cases?
   - Explain the working of your suggested solution by writing code.

Note: You have to properly explain the algorithm. Copy pasting from the internet and plagiarism will lead to straight 0.

Q6:KMP) (5+5)

(a) Given below is an arbitrary pattern:

"aabaabcab"

You have to show the KMP prefix function for the above pattern.

(b) You are required to write a program in C++ which will show how you achieved the KMP prefix function for part A.

Q7:Rabin-Karp) (10 marks)

(a)

For this question, you have to mention the number of spurious hits encountered if we run the Rabin-Karp algorithm on the text given below. Text: 3141592653589793

Assume you are looking for the pattern: P=26, and working modulo: q=11. Note: Make sure you give a detailed description of your work to support your answer.

(b)
You are required to write a program in C++ to code your solution of part A.

Q8) (10 marks)
Discuss how to extend the Rabin-Karp method to handle the problem of looking for a given m x m pattern in an n x n array of characters (The pattern may be shifted vertically and horizontally, but it may not be rotated.)

Q9) (6+10+10)
Part a:
Study a new string matching algorithm of your choice. Make sure you have not studied it in class before. Explain the working of the algorithm by discussing an example.
Note: Plagiarizing the details of the algorithm from the internet will lead to 0 marks. You can search the web but make sure to understand the algorithm and explain it yourself.

Part b:
You are required to write a program in C++ to code your algorithm for part A. Plagiarism from the internet will lead to 0 marks on this question.

Part c:

1. Design a finite automata machine along with state table for string matching that accepts all strings ending the form "ababacb".

2. Prove that string S is a substring of string A by drawing a state diagram of S, and using Finite Automata.

S: a b a b a d c a

i) **A: a b a b a b d c a a b a b a d c a**
ii) **A: a b a b a d c a a b a b a d c a d**
iii) **A: a b a b a b d c b a b a b a d c b**

Answer Format- Answer should be according to the following format:
S: a b
A: b b a b b a b

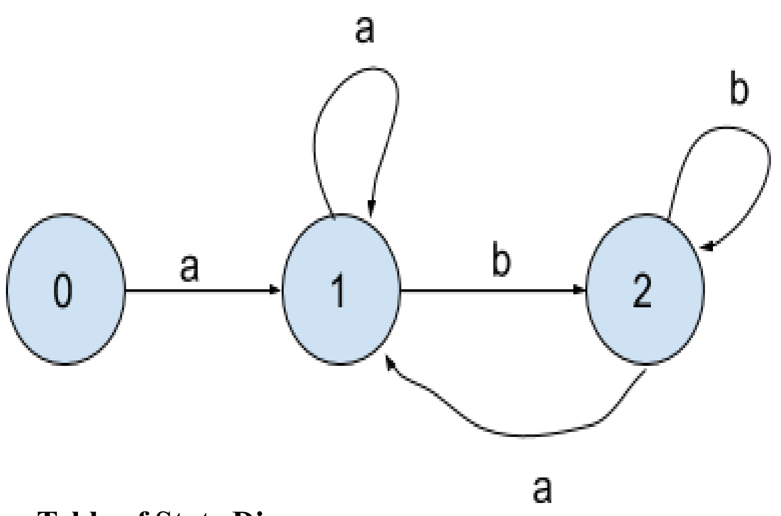

Table of State Diagram:

State	**a**	**b**
0	**1**	**0**
1	**1**	**2**
2	**1**	**2**

Table for solution set:

A	**State**
b	**0**
b	**0**
a	**1**
b	**2 (String matched)**
b	**2**
a	**1**

b	**2(String M)**