Question 1 [4+6=10 Marks]

a) In a d-ary heap, d defines the branching factor of heap. In the class, we discussed the formulae for 2-ary heap (d=2). You are required to describe a 3-ary heap starting with array index 1 by writing the formulae for array indexes to store the children of an i-th node, index of the root node, and the index of the parent of an i-th node. [4 Marks]

Root of the heap is stored at the array index 1
Children of a node i are stored at the array indices **3i-1** to **3i+1**
Parent of a node i is stored at **floor((i+1)/3)**

b) Build Max-Heap for the given set of values: 10, 5, 12, 7, 14, 6, 16, 18, 8, 2, 25. After Max-Heap is formed, perform one Extract_Max_Heap() operation and then draw the final tree and write the final array. [6 Marks]

Final array:

Index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Value	18	16	14	12	7	6	10	5	8	2	~~25~~

Question 2 [5+5+5+5=20 Marks]

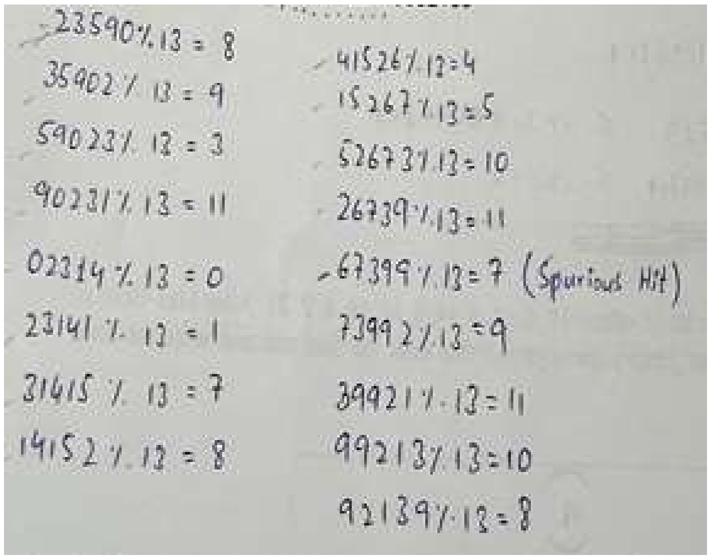
a) Find the spurious hit (Ref: Rabin-Karp algorithm) in the given text string. [5 Marks]

Pattern: 31415

q (for taking modulus): 13

Text: 2359023141526739921 39

Solution: 13-17



23590 % 13 = 8
35902 % 13 = 9
59023 % 13 = 3
90231 % 13 = 11
02314 % 13 = 0
23141 % 13 = 1
31415 % 13 = 7
14152 % 13 = 8

41526 % 13 = 4
15267 % 13 = 5
52673 % 13 = 10
26739 % 13 = 11
67399 % 13 = 7 (Spurious Hit)
73992 % 13 = 9
39921 % 13 = 11
99213 % 13 = 10
92139 % 13 = 8

b) Give a linear-time algorithm to determine whether a text T is a cyclic rotation of another string T′. For example, arc and car are cyclic rotations of each other. You can use the procedures discussed in the class by writing their names and passing appropriate arguments (parameters) without reproducing the whole algorithms. [5 Marks]

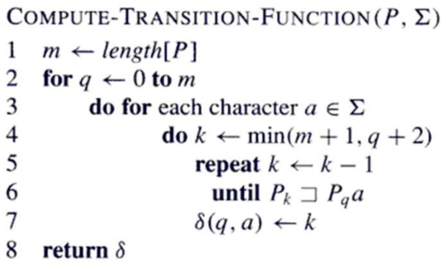
To determine whether a text T is a cyclic rotation of another string T', we can use a simple linear-time algorithm based on string concatenation and substring matching. Here's how it works:

1. Concatenate T with itself to create a new string, T_concat.
2. Check if T' is a substring of T_concat. (Using KMP)
3. If T' is a substring of T_concat and its starting index is within the range of 0 to len(T) - 1, then T is a cyclic rotation of T'. Otherwise, it is not.

```
Procedure check_String_Rotation(T,P,n,m)
BEGIN
  IF (m ≠ n) THEN
           RETURN false
  ENDIF
  IF (there is a valid shift of KMP-MATCHER(TT,P)) THEN
                                               {TT = T concatenated with T}
           RETURN true
  ELSE
           RETURN false
  ENDIF
END
```

Time Complexity: θ(n) because the time complexity of KMP algorithm is θ(n).

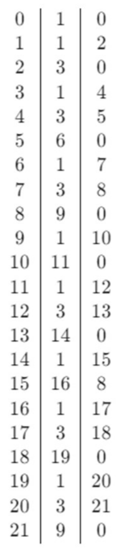
c) Consider the transition table creation algorithm below: [5 Marks]



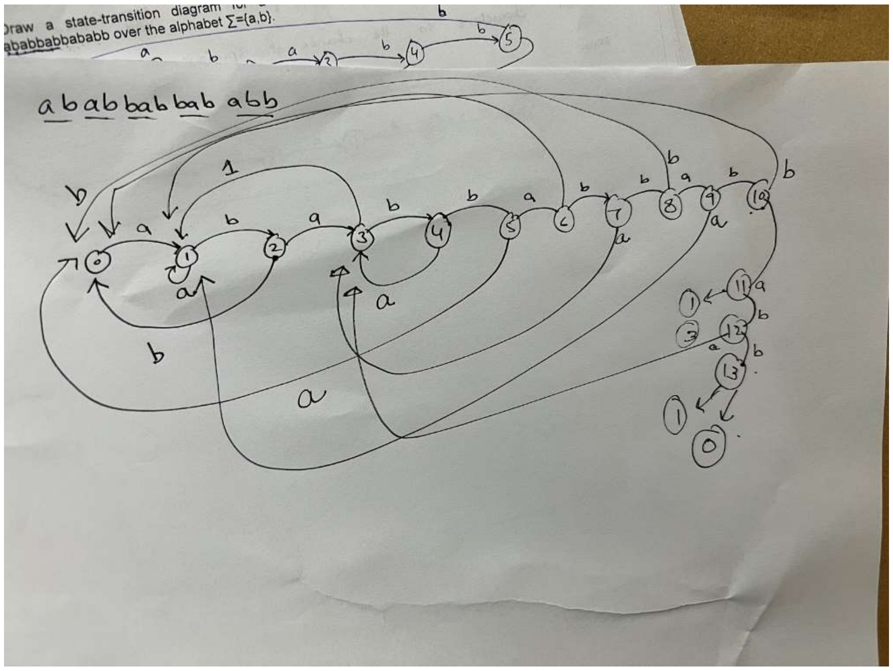
```
COMPUTE-TRANSITION-FUNCTION(P, Σ)
1  m ← length[P]
2  for q ← 0 to m
3       do for each character a ∈ Σ
4              do k ← min(m + 1, q + 2)
5                 repeat k ← k − 1
6                   until P_k ⊐ P_q a
7                 δ(q, a) ← k
8  return δ
```

Draw a state-transition diagram for a string-matching finite-state automaton for the pattern ababbabbababbababbabb over the alphabet ∑={a,b}.

$ababbabbababbababbabb$ over the alphabet $\sigma = \{a, b\}$.



0	1	0
1	1	2
2	3	0
3	1	4
4	3	5
5	6	0
6	1	7
7	3	8
8	9	0
9	1	10
10	11	0
11	1	12
12	3	13
13	14	0
14	1	15
15	16	8
16	1	17
17	3	18
18	19	0
19	1	20
20	3	21
21	9	0


Draw a state-transition diagram for
ababbabbababb over the alphabet Σ={a,b}.
a b a b b a b b a b a b b

d) Suppose that all characters in the pattern P are different. Show how to accelerate naive string matching algorithm to run in time O(n) on an n-character text T by writing the algorithm (in pseudocode form). [5 Marks]

```
Accelerated_naive_algo(T,P,n,m)
BEGIN
          s ← 0
WHILE s ≤ n-m DO
FOR (j : 1 to m) DO
                              IF P[j] ≠ T[s+1] THEN
                                        break
                              END IF

                              IF j = m THEN
                                        PRNT "Pattern occurs at shift : ", s-m
                              END IF

                              s ← s + 1
                    END FOR
          END WHILE
END
```

Question 3 [10+5+5=20 Marks]

a) Consider the following set of activities S={(6,9),(1,10),(2,4),(1,7),(5,6),(8,11),(9,11)}. Select the largest possible number of tasks from S that can be completed without incompatibilities (two activities are incompatible iff they overlap) by using the Greedy-Activity-Selector algorithm learnt in the class. Show all steps (i.e. the complete trace of the algorithm). [10 Marks]

[2,4) then [5,6) then [6,9) then [9,11).

b) Write the time complexity of Prim's algorithm considering that the queue used in the algorithm is a linear array (instead of binary min-heap). Justify your answer. [5 Marks]

$O(V^2 + E) = O(V^2)$. Please consult the lecture slides

c) Write a greedy solution of fractional knapsack problem. Also perform the complexity analysis. Write your algorithm in pseudocode form. [5 Marks]

```
GreedyFractionalKnapsack(items[], capacity)
BEGIN
    Sort items[] by the ratio of value/weight in non-increasing order  {Using Quicksort}
    totalValue = 0
    remainingCapacity = capacity
    FOR each item in items[] AND remainingCapacity ≠ 0 DO
            IF (item.weight <= remainingCapacity)
                    totalValue = totalValue + item.value
                    remainingCapacity = remainingCapacity - item.weight
            ELSE
                    totalValue = totalValue + (remainingCapacity / item.weight) * item.value
                    remainingCapacity = 0
            END IF
    END FOR
    Return totalValue
 END
```

O(nlogn)