General Instructions:

For every subpart the following points are mandatory:

- Recurrence Relation and complexity derivation.
- What each variable stands for
- Pseudo Code
- Name of the algorithm used, and how the answers of subproblems are used to find the answer of the main problem. Also mention the subproblem which will give the final answer.

Question 1.

Mickeymouse loves numbers in the range [m1, m2] (m1 and m2 included). Minnie decides to gift Mickey an array of numbers for his birthday. Mickey wants to find the number of pairs of indices [l, r] for which the sum of all the elements in the range [l, r] lies between m1 and m2.

1. Come up with an algorithm with a worst case complexity of $O(N^2)$. (5 marks)
2. Now improvise this algorithm, assuming all numbers are positive integers. (7 marks)
3. What if the numbers could be negative as well? Can you think of an O(nlogn) solution in this case? (Hint: use sorting) (10 marks)

Sample case:
Array = [-2, 5, -1]
M1 = -2
M2 = 2
Output: 3
[[0, 0], [2, 2], [0, 2]] are the three pairs of [l, r] that satisfy the condition. So, that is why, the output is 3.

Notes: Recurrence relation not needed in first two parts. But do explain the reason for the resulting time complexity in words.
Following points have to be included in your answer:

a) Explanation of algorithm
b) Pseudo Code
c) Justification of complexity of algorithm /Recurrence relation and its solution.

Question 2:

Rahul lives in City A and would like to travel to City B for work. There is a shuttle service for people in these cities which has a fixed schedule. The schedule for City A is a list of boarding times(at City A) and departure times(from City A) for each bus.
Note: No one is allowed to board a shuttle after the boarding time.
He arrives at time t and sometimes has to wait at the station. You are given a list of arrival times for n days.

a. Devise an algorithm to find him the shuttle with the least waiting time. (waiting time = $boarding_j - t$, where j is the next shuttle. And $boarding_j >= t$) for each t_i

b. If he also has access to the travel time of each shuttle. Can you help him find the shuttle which will help him reach his destination at the earliest ?

Return an array of shuttle indexes that Rahul took for n days.

Example Test Case:
Boarding and Departure times of 4 buses travelling from A to B:
[[“0900”,“1000”], [“1100”,“1130”], [“1030”,“1100”], [“0930”,“1030”]]
Rahul’s arrival time for n days:
[“0850”, “0930”, “1021”, “1125”]
A) Index of shuttles: [0,3,2,-1]
Assume extra info : travel Time for 4 shuttles [“2 hrs”, “1 hr”, “1 hr 45 mins”, “1h 20 mins”]
B) Index of shuttles: [3,3,1,-1]

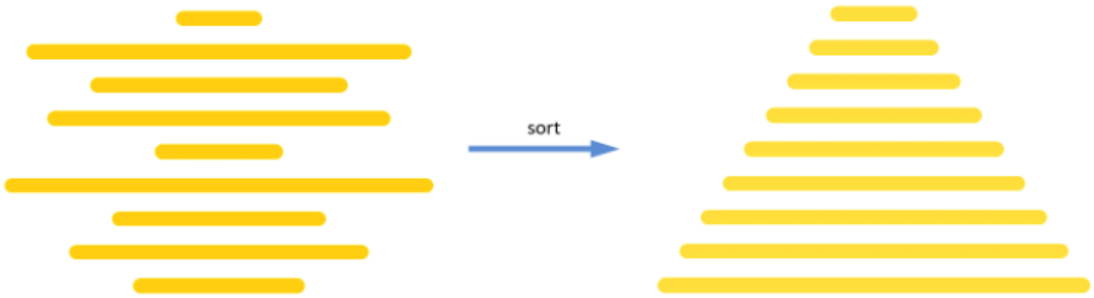
Note:

If there is no such shuttle return -1.
Consider Boarding times and departure times to be arrays with ith element denoting the time of ith bus.
Write your algorithms as pseudo-code and make sure to explain how your algorithm works and the time and space complexity of your algorithm.

Marking: (10 + 10) Total 20 points.

Question 3:

Consider a pile of coins. You have given ‘n’ coins of different sizes and you want to sort the pile so that smaller coins are on the top of larger ones. The only “operation” you are allowed to perform is- take top ‘k’ coins together, and place them back to the pile upside down (See fig for reference).



Example Showing the inversion operation performed on the pile.

a. Describe an algorithm to sort an arbitrary pile of n coins using O(n) operations. Exactly how many operations does your algorithm perform in the worst case.
b. For every positive integer ‘n’, describe the pile structure that requires (Omega) Ω(n) operations to sort. In other words, generate an example of a pile that will require at least cn operations for some c > 0. (n > 3)
c. Now suppose in the sorted array we want all coins to have heads face up. Describe an algorithm to sort an arbitrary pile of ‘n’ coins, such that all coins have tails face down, using O(n) operations. Exactly how many operations does your algorithm perform in the worst case. (No pseudo code required for this subpart, just explain the changes in algorithm and give the worst case analysis)

Note: Assume that the coin information is given to you in the form of an ‘n’ element array A. A[i] is a number between 1 and n and A[i] = j means that the j’th smallest pancake is in position i from the top; in other words A[1] is the size of the top most coin (relative to the others) and
A[n] is the size of the bottommost coin.
operation(A, n) should be assumed to be O(1) can be used directly in pseudo code, it flips first n coins.

Marking: Total 30 points. (15 + 5 + 10)

Question 4:

Akshita loves chocolates, numbers and algorithms. Her friends Ina and Mina gave her a and b chocolates respectively. Given her love for numbers, she likes all the numbers that are divisible by either ‘a’ or ‘b’ or both. She wants to find the nth smallest such number that is divisible by either a or b. Suggest her an algorithm to find the desired number, describe the principle on which the algorithm is based, write the pseudocode and the recurrence relation for the same. Also state which subproblem will give the answer to the original problem.

Expected Complexity → $O(\log(N*\min(A,B)))$.

Assume a and b to be co-prime positive integers.

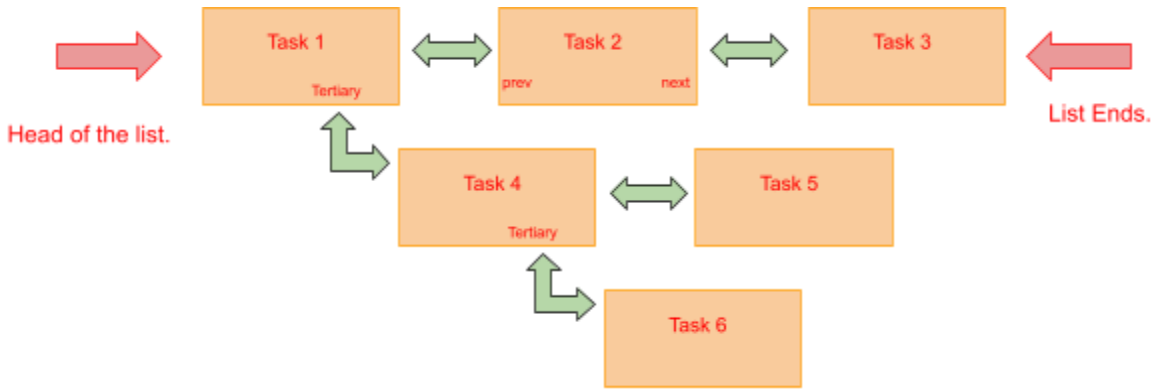
Marking Scheme: Total 15 points.

BONUS Question (10 points)

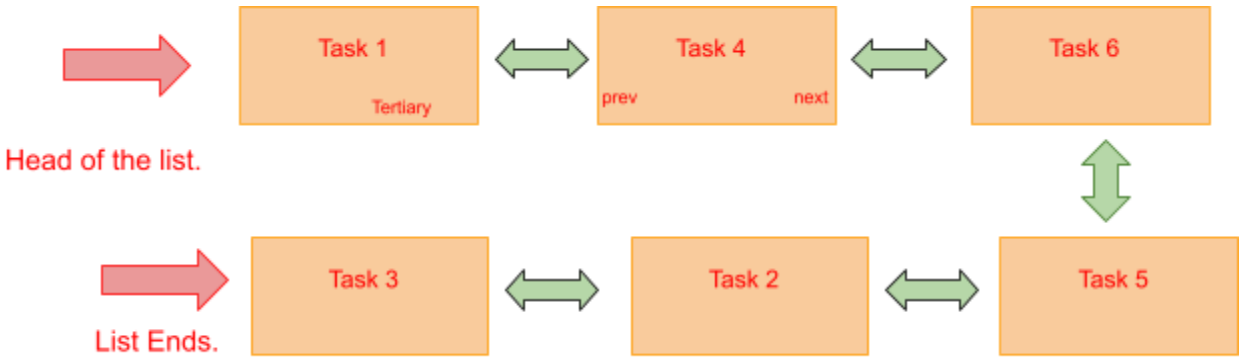
In a hypothetical server, we have a list of tasks arranged in the form of a doubly linked list. In this list in addition to the next and previous pointer each node also has a tertiary pointer, which represents some cleanup tasks required before moving on to the next tasks of the list. The current scheduler however does not recognize this list structure. Your task is to implement an algorithm that takes input the head of the doubly linked list with nodes having a tertiary pointer and gives the head of a rearranged doubly linked list with all tasks arranged linearly (with nodes having just the next and previous pointers only, no tertiary pointer). The modifications should be in-place, that means you cannot change the contents (other than next and previous pointer) or the address of any of the nodes. Also Analyse the running time of the algorithm for the given ‘n’ being the number of tasks.

(For algorithm, write the pseudo code and explain the logic behind it, write base cases and the sub problems too)

Example: Let the given input list be:



The output list that your algorithm should give:


Task 1
Tertiary
Task 4
prev
next
Task 6
Head of the list.
Task 3
Task 2
Task 5
List Ends.