Algorithms Laboratory (CS29203)
Assignment 2: Recursion, Divide & Conquer
Department of CSE, IIT Kharagpur

22nd August 2024

Question-1

(50 points)
Quicksort is an efficient algorithm to sort numbers. However the performance of quicksort gets poor when the input set of numbers contain many repeated elements. For example consider an extreme case where all the input numbers are equal. Then in each recursion iteration, the left partition will remain empty since no numbers are less than pivot, and the right partition will contain most of the elements. This is one extreme case of worst case of quicksort that will take $O(n^2)$ time.

We can improve the algorithm by performing the partition in a clever way. In this case, we can consider three cases: values less than pivot, equal to pivot, and greater than pivot. Since the values equal to pivot are already sorted, only left and right partitions need to be recursively sorted. Your task is to modify the quicksort algorithm to improve its performance for this case. Basically you need to consider three partitions instead of two, and the middle partition is already sorted. You might need to compute the starting and ending indices of the elements having the same value as pivot, instead of a single pivot index as in the traditional quicksort algorithm.

Example:

```
(Input) Enter the number of elements: 10
        Enter the numbers: 2  6  5  2  6  8  6  1  2  6

(Output) 1  2  2  2  5  6  6  6  6  8
```

Question-2

(50 points)
The Autumn semester exam in IIT Kgp is approaching and the institute needs to set-up the examination control mechanisms to have a smooth conduct of exams in Nalanda classroom complex area. One major factor for conducting the exam is to smoothly operate the bus services from main academic building to Nalanda area. There are multiple buses to provide transport services during the exam sessions. The buses run independently of each other, and each bus can continue to have multiple trips successively (that is, the next trip can start immediately after finishing the current trip).

The examination control cell estimates that there needs to be total n number of trips to efficiently conduct the examination process. There is also information available of how much time (in minutes) individual buses take to complete one trip to Nalanda. Let this be denoted as an array *bus[ ]*, where the i-th element in the array denotes the time taken by the i-th bus to complete one trip.

Given the total number of required trips n and the array *bus[ ]*, you have to find the **minimum time** required for all the buses to complete **at least n** number of trips. The complexity of your algorithm should not be worse than $O(k \log(mn))$, where k is the length of the array *bus[ ]* and m is the minimum value in that array. *Hint: Think of binary search algorithm where the left boundary is* 1 *and the right boundary is min(bus[ ]) * n, and then figure out how to proceed further.*

Example 1:

```
(Input) Enter the value of n: 5
        Enter the number of buses: 3
        Enter the array bus[ ]: 1  2  3
(Output) Minimum time is 3

Explanation:
- At time t = 1, the number of trips completed by each bus are [1,0,0].
  The total number of trips completed is 1 + 0 + 0 = 1.
- At time t = 2, the number of trips completed by each bus are [2,1,0].
  The total number of trips completed is 2 + 1 + 0 = 3.
- At time t = 3, the number of trips completed by each bus are [3,1,1].
  The total number of trips completed is 3 + 1 + 1 = 5.
So the minimum time needed for all buses to complete at least 5 trips is 3.
```

Example 2:

```
(Input) Enter the value of n: 1
        Enter the number of buses: 1
        Enter the array bus[ ]: 2
(Output) Minimum time is 2

Explanation:
- There is only one bus, and it will complete its first trip at t = 2.
So the minimum time needed to complete 1 trip is 2.
```