1.

Using Figure 2.4 as a model, illustrate the operation of merge sort on the array $A = \langle 3, 41, 52, 26, 38, 57, 9, 49 \rangle$.

2.

Describe a $\Theta(n \lg n)$-time algorithm that, given a set S of n integers and another integer x, determines whether or not there exist two elements in S whose sum is exactly x.

3. Following code segment is the JAVA implementation of Binary Search,

```
int binarySearch(int arr[], int l, int r, int x)
{
   if (r >= l)
   {
        int mid = l + (r - l)/2;

        if (arr[mid] == x)
            return mid;

        if (arr[mid] > x)
            return binarySearch(arr, l, mid-1, x);

        return binarySearch(arr, mid+1, r, x);
   }

   return -1;
}
```

The time complexity of all divide-and-conquer algorithms can be expressed using following recurrence:

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \le c \, , \\ aT(n/b) + D(n) + C(n) & \text{otherwise} \, . \end{cases}$$

Binary search is a divide-and-conquer algorithm, determine the values of constants a, b, and c, and the orders of growth of $C(n)$ and $D(n)$.

4.

For recurrence $T(n) = \begin{cases} 1, & n = 1 \\ 2T(n/2) + n. & n > 1 \end{cases}$, give the values of T(2), T(4), T(8), T(16) and T(32). Is $T(n) = \mathrm{n}(\lg n + 1)$ true for $n = 1, 2, 4, 8, 16, 32$?

5.

Use the recursion tree to solve recurrence $T(n) = \begin{cases} 1, & n = 1 \\ 5T(n/5) + n. & n > 1 \end{cases}$.