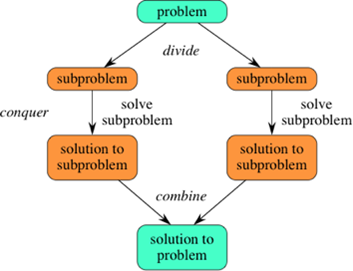
**WRITE NOTES ON**

**1. Explain Divide and Conquer algorithm taking reference to Merge Sort.**

Divide and conquer algorithm first divides the problems into sub problems that are similar to original problem then find the solution to those sub problem recursively, then combine those solution of sub problem to solve the original problem. And each sub problem must be smaller than problem.



Merge sort is a sorting technique based on Divide and Conquer algorithm. In merge sort we have to sort an array. It first divides an array into equal halves which will be sub problems(sub array). Then those sub problems or sub array are sorted and finally are combined to get the solution to the problem, which is the sorted from of an array.

Here first step is to find the mid way of an array a[p..r]. If q be the mid way between p and r then the array is divided into two sub array a[p..q] and a[q+1..r]. Then those two sub arrays are sorted recursively. Then those sorted arrays are combined into a single sorted array a[p..r].

Let’s start with array holding [14,7,3,12,9,11,6,2]

* We can say that array[0..7] where p=0 and r=7
* In the divide step we compute q=3
* The conquer step includes the sorting of two subarrays

array[0..3] = [14,7,3,12]

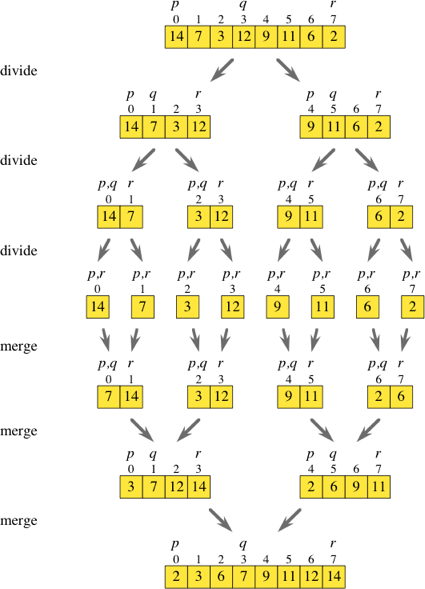
array[4..7]= [9,11,6,2]

* When we comeback from the conquer step, each of the two subarrays is sorted i.e.

array[0..3] = [3,7,12,14]

array[4..7]= [2,6,9,11]

* Finally, the combine step merges the two sorted subarrays in first half and the second half, producing the final sorted array [2, 3, 6, 7, 9, 11, 12, 14].



**2. Write some paragraphs about External Sorting with some examples.**

External sorting is a sorting algorithm that can handle massive amounts of data. External sorting is required when the data being sorted do not fit into the main memory of a computing device (usually RAM) and instead they must reside in the slower external memory (usually a hard drive). External sorting typically uses a hybrid sort-merge strategy. In the sorting phase, chunks of data small enough to fit in main memory are read, sorted, and written out to a temporary file. In the merge phase, the sorted sub files are combined into a single larger file**.**

One example of external sorting is the external merge sort algorithm, which sort the amount of data that only fits on RAM and then merged together after sorting all the data.

For example, for sorting 900 megabytes of data using only 100 megabytes of RAM:

* Read 100 MB of the data in main memory and sort by some conventional method, like quicksort.
* Then write the sorted data to disk.
* Repeat steps 1 and 2 until all of the data is in sorted form, which now need to be merged into one single output file.
* Then those sorted data are merged into a single file.

**3. Explain Partition strategies of Merge Sort and Quick Sort.**

**4. Why is merge sort preferred over quick sort for sorting linked lists?**

Merge sort can work very well with linked list as it doesn’t require additional space. Since we only need to change pointer links rather than shifting the element. In merge sort first we divide the list into number of sub problems having only two elements and then merge them by arranging, so we don’t have to change or swap the elements of whole list we just have to swap elements of sub problems and change the pointer links and then combine them together. But in the quick sort we have to first check whether the element is greater or lesser than other element or not and then we have to change their position and finally we have to change their pointer links too. So merge sort is preferred over quick sort for sorting linked lists.

**5. Write down your analysis on Complexities of each algorithm we've discussed in the class.**

**1. Bubble Sort**

**2. Insertion Sort**

**3. Selection Sort**

**4. Merge Sort**

**5. Quick Sort**

**6. Binary Tree Sort**

**MISSING GAP**

**1. Write something about binary tree sort and analyze the algorithm. WAP a program and submit repo link.**

A tree sort is a sort algorithm that builds a binary search tree from the elements to be sorted, and then traverses the tree (in-order) so that the elements come out in sorted order. Its typical use is sorting elements adaptively: after each insertion, the set of elements seen so far is available in sorted order.

Link for binary tree sort -> <https://github.com/destro014/DSA_Assignment/blob/master/assignment_08/binary_sort.c>