ESC101 : Fundamental of Computing
Lab 11 for 5th November 2008

Note : The solution of the assignment has to be submitted by 5:00 PM on 5th November itself. No late submission is allowed. This is because there is lab test next week and this is the last lab assignment.

1. (Marks = 5)
   We discussed nice examples of recursion in the last few lectures. However, most of them were enumeration problems. For examples, enumerate all strings with n bars and m stars. enumerate all combinations or permutations of a specific length using a set of characters. Recursion is used in various different problems as well. In this lab, you will have to implement a sorting algorithm which employs recursion. This sorting algorithm is called **merge sort**.

   Suppose we are given two sorted arrays A and B. Our aim is to produce third array C which contains the elements of A and B in sorted order. You are not allowed to copy all elements of A and B in to C and then sort the array. Instead you have to somehow use the fact that the arrays A and B are already sorted. Therefore, there should be an easier way to *merge* these two arrays to form C. This problem was asked in the lab 9 held on 21st October. You may have a look at the problem and its solution which is also available.

   Following is pseudocode of the **merge sort** algorithm. The method `merge_sort(A,left,right)` will sort the portion of the array `A` from `A[left],...A[right]`.

```
merge_sort(A, left, right)
{  if(left == right) nothing to be done.
   else
   {    mid = (left+right)/2;
        merge_sort(A,left,mid);
        merge_sort(A,mid+1,right);
        Now merge the two portions
               A[left],...,A[mid]  and  A[mid+1],...A[right]
        and store it in some temporary array Temp.
        Copy the sorted array Temp[] to A[left],...,A[right]
   }
}
```

   You have to write a JAVA program for merge sort. Your program should be interactive : It should first ask the user for the size of the array. The user should enter a positive integer for the size. The program should then create the array of that size. It should then ask user to supply the numbers to be stored in A one by one. After that, it should sort them using **merge sort** and print them on the terminal.

2. (marks=3)
   Suitably change the program designed above so that whenever method `merge_sort()` is called, it should print the parameters `(left, right)` of the method on the terminal. Run your program for arrays of size $3, 5, 7, 10$. This output should provide some insight into the execution of the recursive merge sort algorithm.

3. (Marks = 2)
   You have to adapt the merge sort program designed above with file input and output so that the program should read the size of array A and its numbers from a specific file. It should then print these numbers in sorted order in another file. The sample program for reading from a file and writing into a file will be discussed tomorrow in the Lecture class. This part of the assignment will be very easy.