National University of Computer and Emerging Sciences



**Laboratory Manual**

***(Computer Programming)***

|  |  |
| --- | --- |
| Course Instructor | Sarim Baig |
| Lab Instructor(s) | Ahmad Raza  Waqas Manzoor |
| Section | A, B |
| Semester | Spring 2017 |
| Date | 18-April-2017 |

Department of Computer Science

FAST-NU, Lahore

**Lab Manual (12)**

**(Recursion)**

These methods are required to be **fully recursive** in that they contain **no loops at all;**

neither for, while or do-while.

**1.** *void* ***listNumbers(int start, int end)***that outputs the numbers from *start* to *end* to console. Write one version that outputs the numbers in ascending order, and another that outputs them in descending order.

**2.** ***int min(int[] a, int start, int end)***that returns the smallest element between the indices *start* and *end* in the parameter array a.

**3. *int mul(int a, int b)***that computes the product of two integers *a* and *b*. The only arithmetic

operation that you are allowed to use in this problem is addition +.

**4. Use the sorted list taken as user input. Ask user for an integer value to input. Search this value from the array using recursive binary search. The algorithm is given below.**

1. Find the midpoint of the array; this will be the element at *arr[size/2].* The midpoint divides the array into two smaller arrays: the lower half of the array consisting of elements 0 to midpoint - 1, and the upper half of the array consisting of elements midpoint to size - 1.
2. Compare *key* to *arr[midpoint]* by calling the user function *cmp\_proc*.
3. If the key is a match, return *arr[midpoint];* otherwise
4. If the array consists of only one element return NULL, indicating that there is no match; otherwise
5. If the key is less than the value extracted from *arr[midpoint]* search the lower half of the array by recursively calling *search;* otherwise
6. Search the upper half of the array by recursively calling *search.*

**5. *Permutations of a given string***

Following is the recursive tree for printing all permutations of string “ABC”. Write a recursive function void permute(char \*a, int l, int r). You should also write a function void swap(char \*x, char \*y) to accomplish this task.

/\* Function to print permutations of string

This function takes three parameters:

1. String

2. Starting index of the string

3. Ending index of the string. \*/

