Practical 07
Advanced Sort – Merge Sort

1. Merge Sort

Background: *In computer science, Merge Sort (also commonly spelled Mergesort) is an efficient, general-purpose, comparison-based sorting algorithm. Most implementations produce a stable sort, which means that the order of equal elements is the same in the input and output. Merge Sort is a divide and conquer algorithm that was invented by John von Neumann in 1945. A detailed description and analysis of bottom-up Merge Sort appeared in a report by Goldstine and von Neumann as early as 1948.*

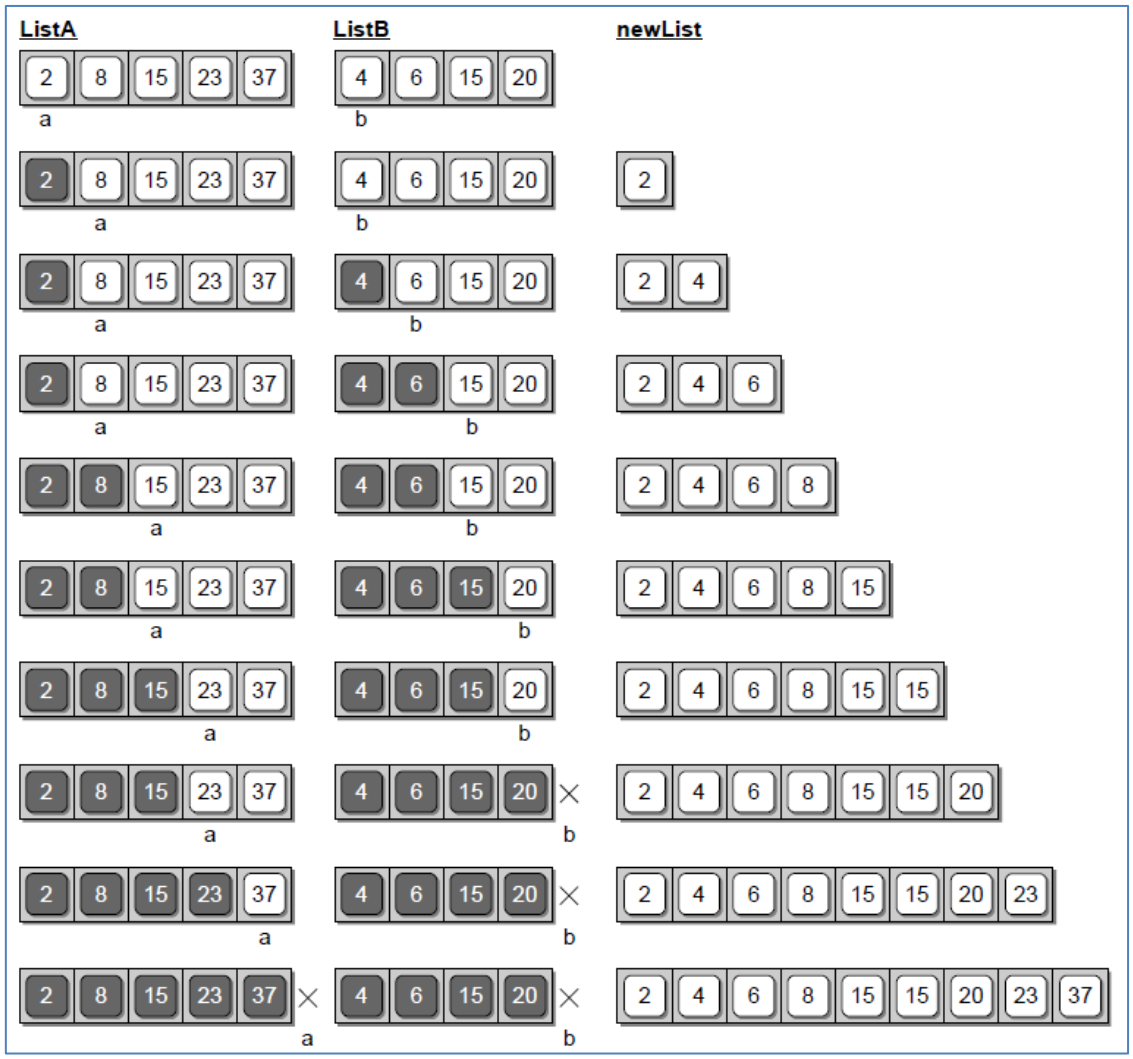
(Reference: https://en.wikipedia.org/wiki/Merge_sort)

The following code is a partial implementation of the Merge Sort algorithm using recursion:

```
# Sorts a Python list in ascending order using the merge sort
# algorithm
def mergeSort( theList ):

    # Check the base case - the list contains a single item
    if len(theList) <= 1:
        return theList
    else:
        # Compute the midpoint
        mid = len(theList) // 2

        # Split the list and perform the recursive step
        leftHalf = mergeSort( theList[ :mid ] )
        rightHalf = mergeSort( theList[ mid: ] )

        # Merge the two sorted sublists
        newList = mergeSortedLists( leftHalf, rightHalf)
        return newList
```

The implementation of the `mergeSortedLists()` function is not included above. Basically, the function takes in two sorted lists (`leftHalf` and `rightHalf`) and merge them to create a new sorted list.

A possible approach to implement `mergeSortedLists()` is provided by the reference text – *Data Structures & Algorithms using Python. Rance D. Necaise, Wiley, 1st Edition, 2011* , using the illustration depicted below.



NOTE: **a** and **b** are index variables indicating the next value to be merged from the respective list.

Based on the approach illustrated above, complete the implementation of the `mergeSortedLists()` given below, by providing the rest of the required code.

```
# Merge two sorted lists to create and return a new sorted list
def mergeSortedLists( listA, listB ):

    # Create the new list and initialise the list markers
    newList = __________
    a = __________
    b = __________

    # Merge the two lists together until one is empty
    while a < len( listA ) and b < len( listB ):
        if listA[a] < listB[b]:
            _____________________________
            _____________________________
        else:
            _____________________________
            _____________________________

    # If listA contains more items, append remaining items to
    # newList
    while _____________________:
        _____________________________
        _____________________________

    # If listB contains more items, append remaining items to
    # newList
    while _____________________:
        _____________________________
        _____________________________

    return newList


# Test code
list_of_numbers = [12, 7, 9, 24, 7, 29, 5, 3, 11, 7]

print('Input List:', list_of_numbers)
merge_list = mergeSort(list_of_numbers)
print('Sorted List:', merge_list)
```

Sample Output:

```
Input List: [12, 7, 9, 24, 7, 29, 5, 3, 11, 7]
Sorted List: [3, 5, 7, 7, 7, 9, 11, 12, 24, 29]

Process finished with exit code 0
```

- End of Practical --