CS2302 - Data Structures

Spring 2019

Lab 2

Due Friday, February 22, 2019

The median of a list L is the element a such that half of the elements in L are smaller than a and half of them are larger than a. For example, the median of list $L = \{20, 10, 45, 1, 12\}$ is 12, since 12 is grater than 1 and 10 and smaller that 20 and 45.

An easy way to find the median is to sort the list and return the element in the middle:

```
def Median(L):
    C = Copy(L)
    Sort(C)
    return ElementAt(C,GetLength(C)//2)
```

Your task for this lab is to implement several algorithms for finding the median of a list of integers, using objects of the *List* class described in class, and compare their running times (measured as the number of comparisons each algorithm makes) for various list lengths. To generate data to test your methods, write a method that receives an integer n and builds and returns a list of random integers of length n.

The algorithms to compare are the following:

1. Sort list using bubble sort, then return the element in the middle.
2. Sort list using merge sort, then return the element in the middle.
3. Sort list using quicksort, then return the element in the middle.
4. Implement a modified version of quicksort that makes a single recursive call instead of the two made by normal quicksort, processing only the sublist where the median is known to reside.

Write a report describing your work. For every method, determine the big-O running time with respect to n. Run experiments with various values of n and determine the number of comparisons that each algorithm makes and determine if their analytical running times agree with what you see in practice. Illustrate your experimental results using tables and/or plots. Make sure that for a given array all 4 algorithms return the same value.