CS2023 - Data Structures and Algorithms
In-class Lab Exercise

Week 5

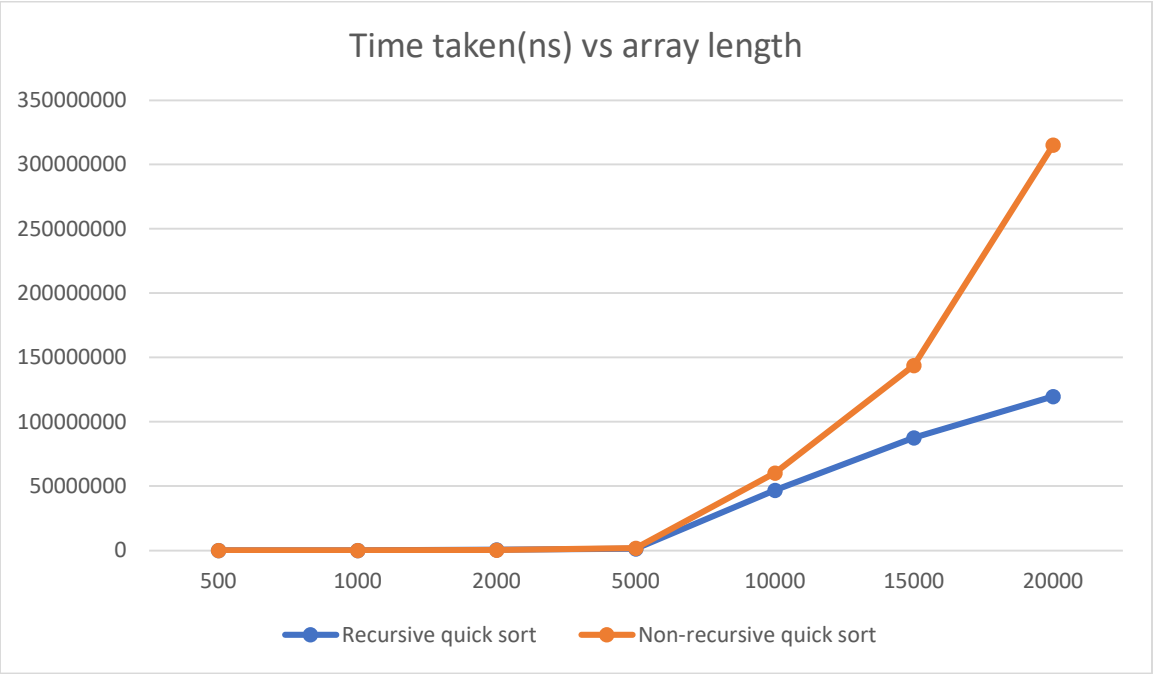
Index: 200381U

Question 1

- Implement Recursive Quick Sort algorithm.
- Watch the following video to get to know how does quick sort work.
    - https://www.youtube.com/watch?v=Vtckgz38QHs&ab_channel=BroCode (Watch until 7.45 minutes)
- Try to implement non-recursive quick sort algorithm.
- Compare time taken for execution and plot the graph.

Answer 1

```
Total elements in array: 500
Time taken by recursive approach: 109441 nanoseconds
Time taken by iterative approach: 90378 nanoseconds
Total elements in array: 1000
Time taken by recursive approach: 212566 nanoseconds
Time taken by iterative approach: 164001 nanoseconds
Total elements in array: 2000
Time taken by recursive approach: 480870 nanoseconds
Time taken by iterative approach: 470414 nanoseconds
Total elements in array: 5000
Time taken by recursive approach: 1420823 nanoseconds
Time taken by iterative approach: 1908817 nanoseconds
Total elements in array: 10000
Time taken by recursive approach: 46933922 nanoseconds
Time taken by iterative approach: 60266139 nanoseconds
Total elements in array: 15000
Time taken by recursive approach: 87563238 nanoseconds
Time taken by iterative approach: 143850694 nanoseconds
Total elements in array: 20000
Time taken by recursive approach: 119825698 nanoseconds
Time taken by iterative approach: 315206747 nanoseconds
```



Comparison: Up until array length is 5000 both algorithms have taken roughly same amount of time. Non-recursive algorithm has shown better performance when sorting these small arrays. But as array length increases, time taken by non-recursive approach is increasing rapidly while time taken by recursive approach increases linearly. It is quite obvious from our observations that recursive algorithm has a better performance when considering time factor of sort large arrays.

Question 2: Find running median.

- The median of a set of integers is the midpoint value of the data set for which an equal number of integers are less than and greater than the value. To find the median, you must first sort your set of integers in non-decreasing order, then:
    - If your set contains an odd number of elements, the median is the middle element of the sorted sample. In the sorted set {1, 2, 3}, 2 is the median.
    - If your set contains an even number of elements, the median is the average of the two middle elements of the sorted sample. In the sorted set {1, 2, 3, 4}, (2+3)/2=2.5 is the median.
- Given an input stream of n integers, perform the following task for each i^{th} integer: ○ Add the i^{th} integer to a running list of integers.
    ○ Find the median of the updated list (i.e., for the first element through the i^{th} element). ○ Print the updated median on a new line. The printed value must be a double-precision number scaled to 1 decimal place (i.e., 12.3 format).
- Example:
    ○ *Input*: a = [7, 3,5, 2] ○
    Output:

Sorted	Median
[7]	7.0
[3, 7]	5.0
[3, 5, 7]	5.0
[2,3,5,7]	4.0

Answer:

Test 1 -

```
 array length is 8
78
sorted array : 78  ||  The median is : 78
89
sorted array : 78,89  ||  The median is : 83.5
755
sorted array : 78,89,755  ||  The median is : 89
89
sorted array : 78,89,89,755  ||  The median is : 89
84
sorted array : 78,84,89,89,755  ||  The median is : 89
78
sorted array : 78,78,84,89,89,755  ||  The median is : 86.5
45
sorted array : 45,78,78,84,89,89,755  ||  The median is : 84
44
sorted array : 44,45,78,78,84,89,89,755  ||  The median is : 81
```

Test 2 –

```
10
 array length is 10
45
sorted array : 45  ||  The median is : 45
789
sorted array : 45,789  ||  The median is : 417
854
sorted array : 45,789,854  ||  The median is : 789
555
sorted array : 45,555,789,854  ||  The median is : 672
864
sorted array : 45,555,789,854,864  ||  The median is : 789
126
sorted array : 45,126,555,789,854,864  ||  The median is : 672
645
sorted array : 45,126,555,645,789,854,864  ||  The median is : 645
895
sorted array : 45,126,555,645,789,854,864,895  ||  The median is : 717
645
sorted array : 45,126,555,645,645,789,854,864,895  ||  The median is : 645
5003
sorted array : 45,126,555,645,645,789,854,864,895,5003  ||  The median is : 717
```

```cpp
#include <iostream>
using namespace std;

int main(){
    int n;
    cin >> n;
    cout << " array length is " << n << endl;
    int arr[n];
    int pointer = -1;
    while(pointer < n-1){
        if(pointer==-1){
            pointer++;
            cin >> arr[pointer];
        }
        else{
            pointer++;
            int current;
            cin >> current;
            int temp_pointer = pointer-1;
            while(temp_pointer>=0){
                if(arr[temp_pointer]>= current){
                    arr[temp_pointer+1]=arr[temp_pointer];
                    temp_pointer = temp_pointer-1;
                }
                else{
                    break;
                }
            }
            arr[temp_pointer+1]=current;
        }
        cout << "sorted array : ";
        for(int i=0;i<=pointer;i++){
            cout << arr[i];
            if(i==pointer) break;
            cout << ",";
        }
        double median;
        if(pointer%2==0) median = (double) arr[pointer/2]/(double) 1;
        else median = (double)(arr[pointer/2]+arr[pointer/2 +
1])/(double)2;

        cout << "  ||  The median is : " << median ;
        cout << endl;

    }
}
```