EL9343 Homework 04

Spring 2022

Name: Sagar Patel
NETID: SP5894

Question 01

For the following array –
$A = < 15, 22, 25, 27, 12, 19, 23, 16, 24, 14, 10, 26 >$

1.a

Create a max heap using the algorithm BUILD-MAX-HEAP.

Solution

The max heap is as shown –

```
[26, 24, 25, 22, 14, 19, 23 16, 17, 12, 10, 15]
```

1.b

Design an algorithm to create a min heap. (Pseudocode is required)

Solution

Function 1: `BUILD-MIN-HEAP()`

```
BUILD-MIN-HEAP(A)
    n = length(A)
    for i which decrements from (n/2) to 1:
        perform MIN-HEAP(A, i, n)
```

Function 2: `MIN-HEAP()`

```
MIN-HEAP(A, i)
    l = Left(i)
    r = Right(i)
    if (l<=heapSize(A) and A[l] < A[i]):
        smallest = l
    else:
        smallest = i

    if (r<=heapSize(A) and A[r] < A[largest]):
        smallest = r

    if (smallest!=i):
        swap A[i] with A[smallest]
        MIN-HEAP(A, smallest)
```

1.c

Create a min heap using the algorithm you designed in 1(b)

Solution

The min heap is as shown —

```
[10, 12, 19, 16, 14, 25, 23, 17, 24, 15, 22, 26]
```

1.d

Remove the largest item from the max heap you created in 1(a), using the `HEAP-EXTRACT-MAX` function. Show the array after you have removed the largest item.

Solution

After removing the largest item —

```
[25, 24, 23, 22, 14, 19, 15, 16, 17, 12, 10, 11]
```

1.e

Using the algorithm `MAX-HEAP-INSERT` , insert 11 into the heap that resulted from question 1(d). Show the array after insertion.

Solution

After insertion —

```
[25, 24, 23, 22, 14, 19, 15, 16, 17, 12, 10, 11]
```

Question 02

Design two different algorithms to merge k sorted arrays, and return it as a new array. The new array should be made by splicing together the nodes of the k arrays. Additionally, the total number of elements of all arrays is kn. (Notice that the number of elements of each array is not necessary the same). One of your algorithms should run in $O(kn.logn)$ time. Please give the procedure of your algorithm and analyze the running time. (Description is enough, you do not need to provide any pseudocode)
Input A: $< 1, 4, 7, 10 >$, B: <2,5,8,11>, C:<3,6,9>
Output: $< 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 >$

Solution

We can initialize a minHeap and insert the first element in all the arrays and that into the heap. The cost of this will be $O(k)$. Now, we repeat the underlying operations until the arrays have been scanned till the last item −

- Get the smallest element from the heap using EXTRAC-MIN and store it in the output array.
- Replace the heapRoot with the next element from the array where the element is extracted. If the array does not have any more elements then replace the root with an undefined number (infinity). Once that is done, call the HEAPIFY(A,1) function.

That would be because since we have kn keys, the `HEAPIFY` takes approximately $O(logk)$ time. Therefore, the total running time would be $O(k) + kn.O(logk) = O(kn.logk)$

Any other algorithm can be implemented as −

1. Merge individually (one by one).
2. Merge in pairs.
3. Combine them all and sort.

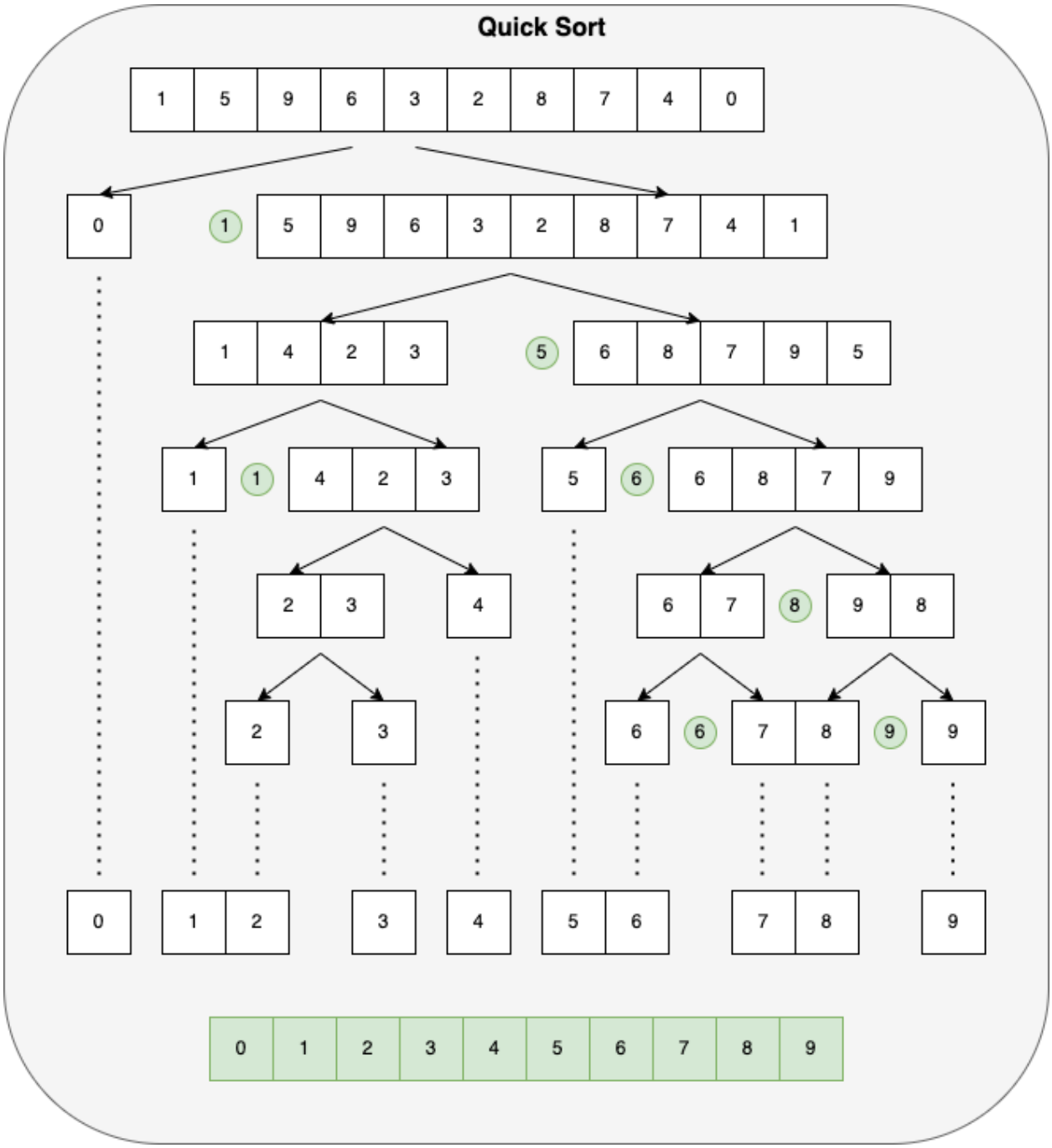
Question 03

For the following array $A = < 1, 5, 9, 6, 3, 2, 8, 7, 4, 0 >$

3.a

Illustrate the operation of quick sort on array A.

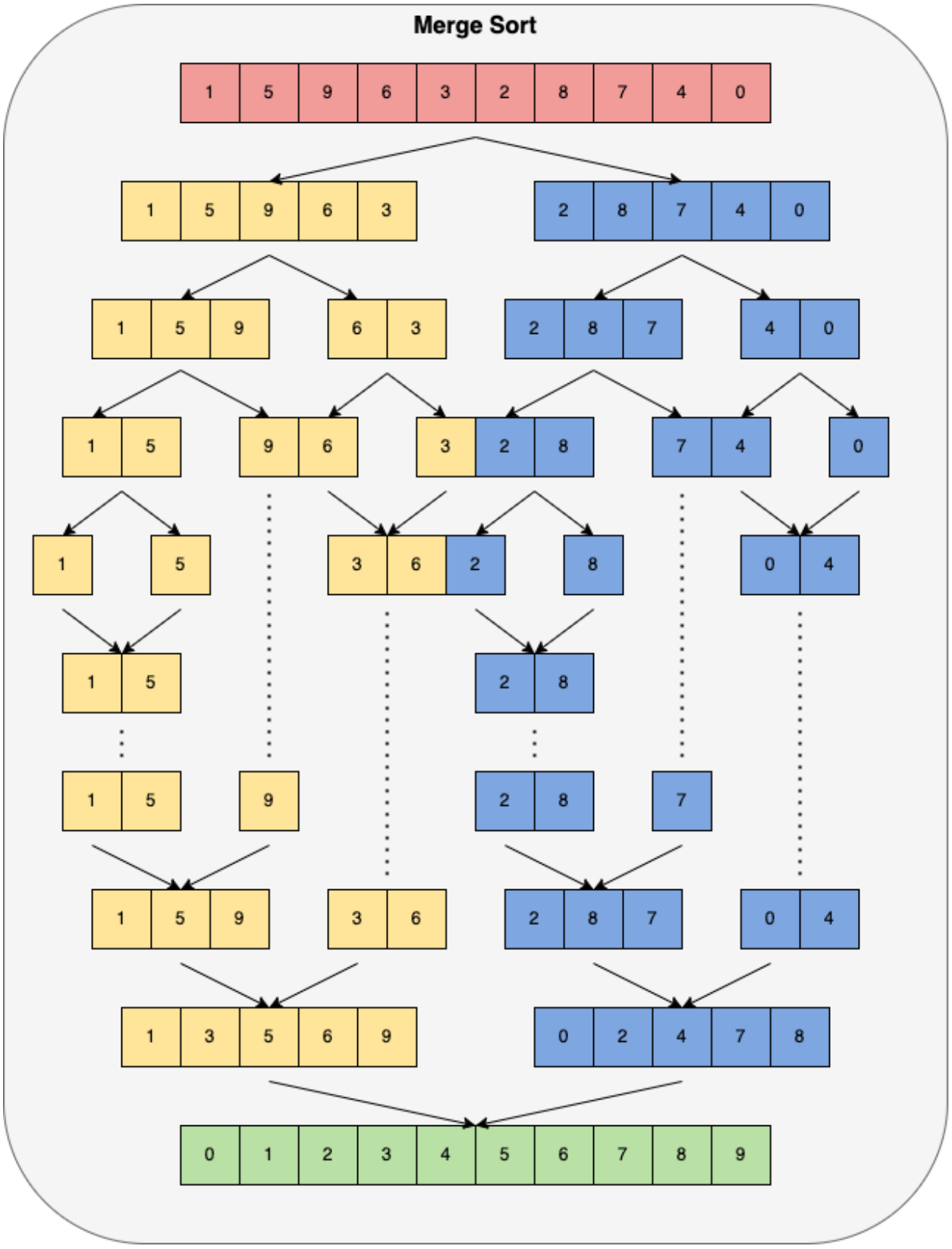
Solution



3.b

Illustrate the operation of merge sort on array A.

Solution



3.c

Explain the advantage and disadvantage of sorting an array by quick sort compared to using merge sort.

Solution

They can be compared as follows —

	Quick Sort	Merge Sort
Worst Case	$O(n^2)$	$O(n.\ logn)$
Average Case	$O(n.\ logn)$	$O(n.\ logn)$
Stability	Unstable	Stable
Storage	in-place	$O(n)$

Question 04

For an disordered array with n elements, design an algorithm for finding the median of this array. Your algorithm should traverse the array only once.

Solution

The algorithm is as follows —

```
MEDIAN(A)
    heapSize = Size(A)
    heap = buildMinHeap(A,0,heapSize)
    for i in range of (heapSize+1) to Size(A):
        if A[i] > heap[1]:
            swap heap[1] with A[i]
                minHeapify(heap, 1)
    if Size(A)%2 == 0:
        return heap[1]
    else:
        return (heapExtractMinimum + heap[1])/2
```