CSE100 Algorithm Design and Analysis Discussion Ch 08 Spring 2022

You're expected to work on the problems before coming to the lab. Discussion session is not meant to be a lecture. TAs will guide the discussion and correct your solutions if needed. We may not release solutions for all problems. If you're better prepared for discussion, you will learn more. TAs will record names of the students who actively engage in discussion and report them to the instructor. The instructor will factor in participation in final grade.

1. (Intermediate) Draw the decision tree for Merge sort operating on three element; so Merge-sort(A, 1, 3) where the input is $A[1...3] = \langle a_1, a_2, a_3 \rangle$. Use the decision tree for Insertion sort in Fig. 8.1 as a model. Note that each internal node must correspond to a comparison of two given elements from $A[1...3]$.
   Sol. See the handwritten jpg file.

2. (Basic) What is a stable sort?
   Sol. elements with the same key must appear in the same order as they did before the sorting.

3. (Basic) Explain why the decision tree for any comparison based sorting algorithm must have at least $n!$ leaves.
   Sol. There are $n!$ possible outcomes and each of them must appear in one of the leaves.

4. (Basic) Show $\log_2 n! = \Omega(n \log n)$.
   Sol. For simplicity, assume that n is even. $n! \geq (n/2+1)(n/2+2)...(n/2+n/2) \geq (n/2)^{n/2}$.

5. (Basic) Illustrate the operation of Counting-Sort on $A = \langle 6, 0, 2, 0, 1, 3, 4, 6, 1, 3, 2 \rangle$.
   Sol. Omitted.

6. (Advanced) Describe an algorithm that, given n integers in the range from 0 to k, preprocesses the input and then answers any query on how many of the n integers fall into range $[a...b]$ in $O(1)$ time. Your algorithm should use $O(n+k)$ preprocessing time.
   Sol. Following the counting sort, we can compute $C[0...k]$ such that $C[i]$ is set to the number of elements no greater than i. Then, given a query, we just need to return $C[b]-C[a-1]$; here $C[-1] = 0$.

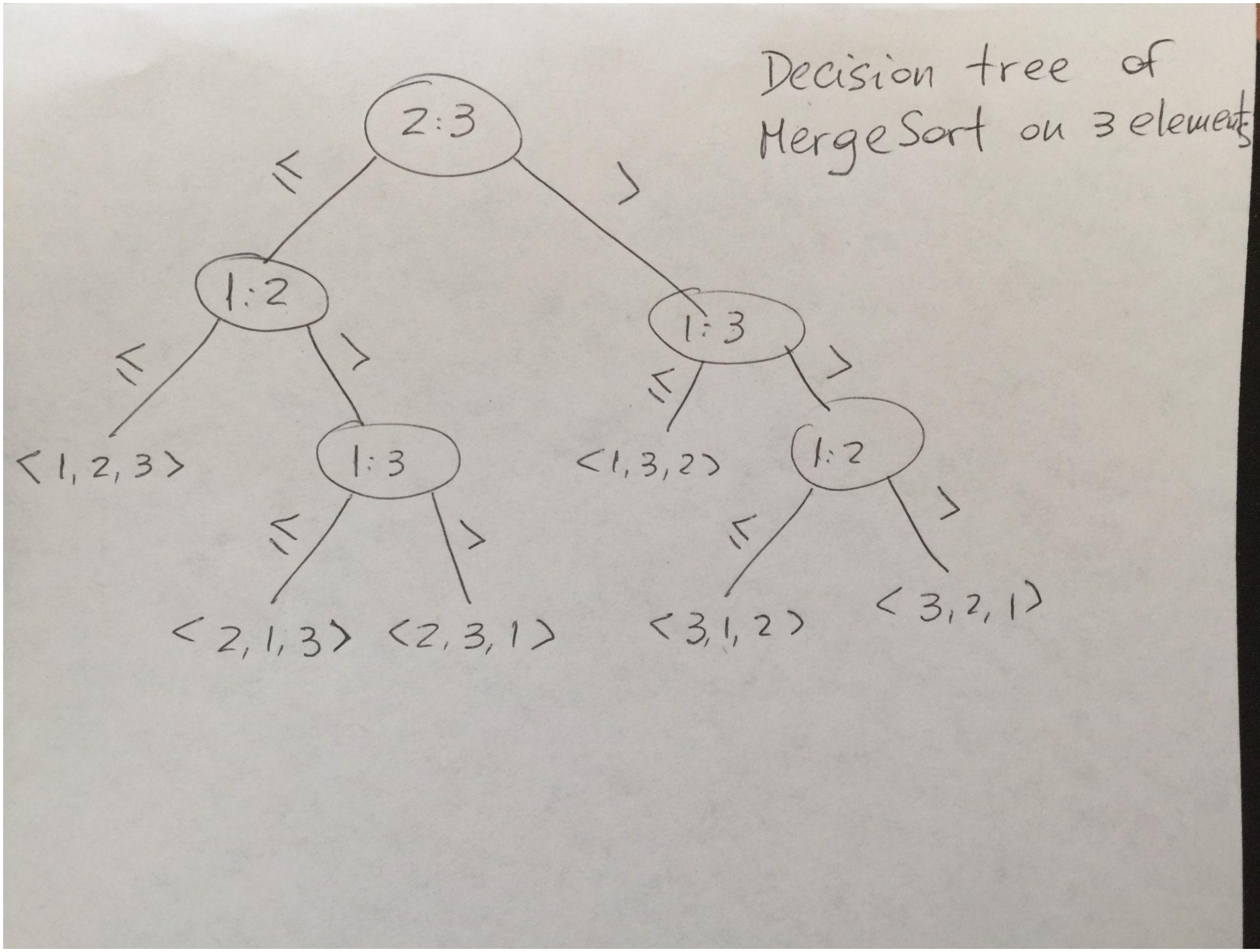
7. (Intermediate) Which of the following sorting algorithms are stable: insertion sort, merge sort, and quicksort (according to their implementations in the textbook)?
   Sol. Insertion, Merge. But not quicksort.

8. (basic) Using Figure 8.4 as a model, illustrate the operation of Bucket-Sort on the array $A = \langle .79, .13, .16, .64, .39, .20, .89., 53., .71, .42 \rangle$.
   Sol. See the textbook or lecture slides.

9. (intermediate) Explain why the worst-case running time for bucket sort is $\Theta(n^2)$? What simple change to the algorithm preserves its linear average-case running time and makes its worst-case running time $O(n \log n)$?
   Sol. All elements could be assigned to the same bucket. Then, sorting the linked list of n elements via insertion sort could take $\Omega(n^2)$ time. A simple fix is to sort the list using Merge sort or heap sort to, which will improve the running time to $O(n \log n)$.

10. (advanced) Describe a (worst-case) linear time algorithm that decides if a given sequence of integers $\langle a_1, a_2, ..., a_n \rangle$ is a permutation of $\langle 1, 2, 3, ..., n \rangle$. Your algorithm only needs to say Yes or No. (Hint: counting sort.)
    Sol. Omitted.

5)


Decision tree of MergeSort on 3 elements
2:3
≤
>
1:2
1:3
≤
>
≤
>
<1, 2, 3>
1:3
<1, 3, 2>
1:2
≤
>
≤
>
<2, 1, 3>
<2, 3, 1>
<3, 1, 2>
<3, 2, 1>

7)

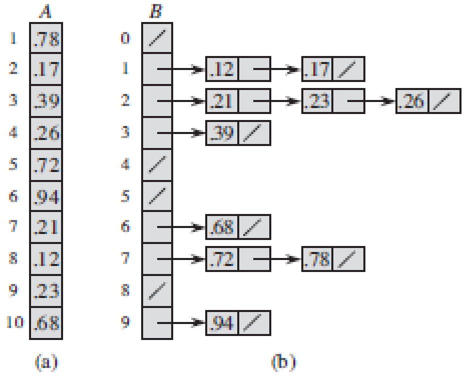


Figure 8.4 The operation of BUCKET-SORT for $n = 10$. **(a)** The input array $A[1..10]$. **(b)** The array $B[0..9]$ of sorted lists (buckets) after line 8 of the algorithm. Bucket i holds values in the half-open interval $[i/10, (i+1)/10)$. The sorted output consists of a concatenation in order of the lists $B[0], B[1], \ldots, B[9]$.

```
BUCKET-SORT(A)
1  let B[0..n − 1] be a new array
2  n = A.length
3  for i = 0 to n − 1
4      make B[i] an empty list
5  for i = 1 to n
6      insert A[i] into list B[⌊nA[i]⌋]
7  for i = 0 to n − 1
8      sort list B[i] with insertion sort
9  concatenate the lists B[0], B[1], ..., B[n − 1] together in order
```

8)

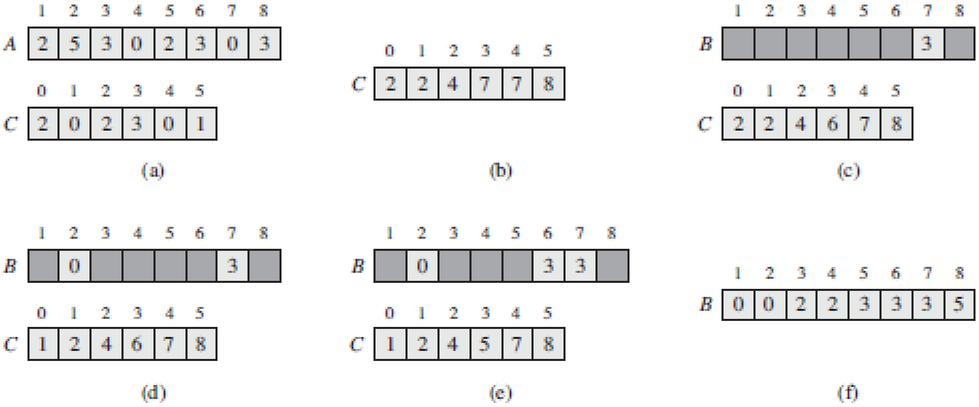
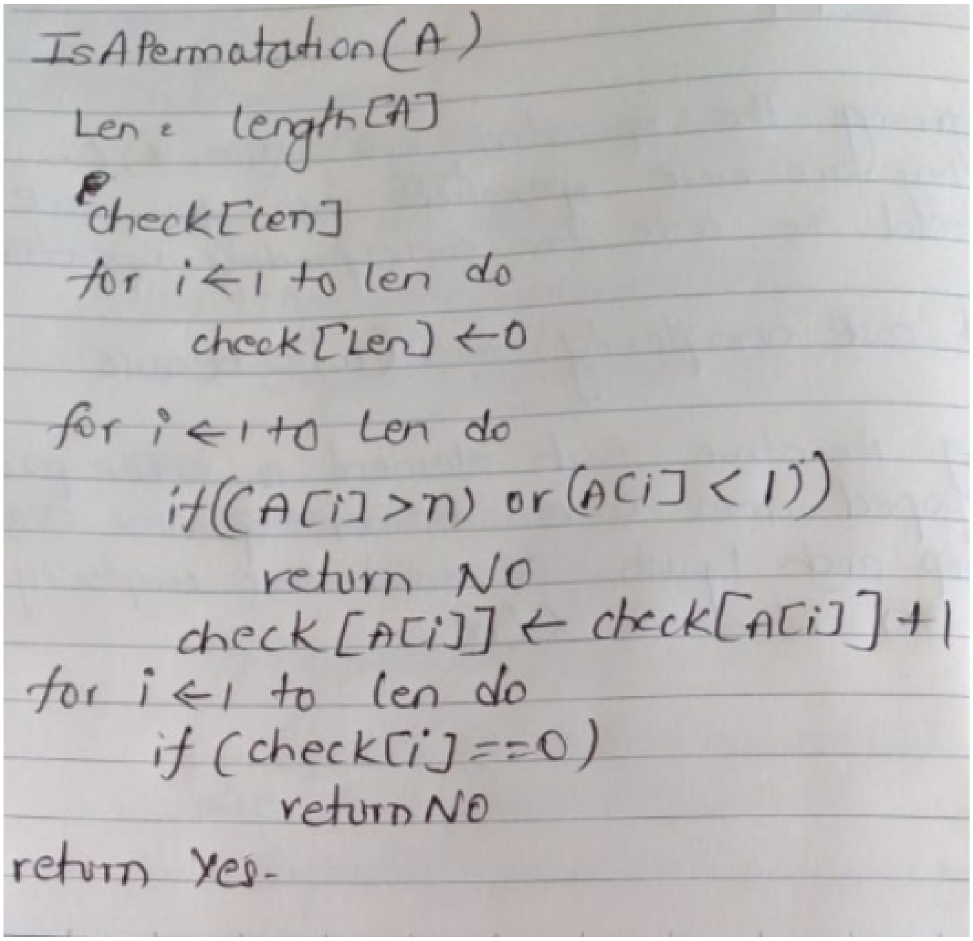


Figure 8.2 The operation of COUNTING-SORT on an input array $A[1..8]$, where each element of A is a nonnegative integer no larger than $k = 5$. **(a)** The array A and the auxiliary array C after line 5. **(b)** The array C after line 8. **(c)–(e)** The output array B and the auxiliary array C after one, two, and three iterations of the loop in lines 10–12, respectively. Only the lightly shaded elements of array B have been filled in. **(f)** The final sorted output array B.

```
COUNTING-SORT(A, B, k)
1  let C[0..k] be a new array
2  for i = 0 to k
3      C[i] = 0
4  for j = 1 to A.length
5      C[A[j]] = C[A[j]] + 1
6  // C[i] now contains the number of elements equal to i.
7  for i = 1 to k
8      C[i] = C[i] + C[i − 1]
9  // C[i] now contains the number of elements less than or equal to i.
10 for j = A.length downto 1
11     B[C[A[j]]] = A[j]
12     C[A[j]] = C[A[j]] − 1
```

10)



```
IsAPermatation(A)
  Len = length[A]
  check[len]
  for i ← 1 to len do
      check[Len] ← 0

  for i ← 1 to Len do
      if((A[i] > n) or (A[i] < 1))
          return NO
      check[A[i]] ← check[A[i]] + 1
for i ← 1 to len do
    if (check[i] == 0)
        return NO
return Yes.
```

```
STEP1: start

STEP2: read an array of n elements i.e, a1,a2,a3,.........,,an.

STEP3: largestelement = a[1]

STEP4: for i = 1 to n repeat step-5

STEP5: if(a[i]<a[i+1] then largestelement= a[i+1] , index1=i+1

STEP6:for i= 1 to n

if(largestelement==a[i] then index2=i

STEP7:if(index1>index2)

then display largest number first occured in index2 position and go to step9

STEP8:display largest number first occured in index1 position.

STEP9: stop
```

WORST CASE TIME COMPLEXITY:

the worst case time complexity is that when the largest element is present at the last position in array

in our algorithm we are using two for loops, which are linear .

so the worst case time complexity is ***2O(n)***

B_n = {1,2,3,...,n}

Then $\bigcup_{n \in Z} B_n$ = Z^+, the set ofpositive integers.

For clearly $B_n \subset Z$ for all integer n. So $\bigcup_{n \in Z} B_n \subset Z^+$.

Let k ε Z^+ then n ε$B_k \subset \bigcup_{n \in Z} B_n$ and therefore $Z^+ \subset \bigcup_{n \in Z} B_n$.

Note 1 is the only common point for allB_n.
Since for any other integer k we have that k>1, does not belongto B_{k-1})
So the intersection of elements in B is {1}.

Similarly we can prove b) also:
Clearly the union of all elements in A is{1,2,3,...,19}
But this finite family of sets has no common element.
Because 1 is not in A_2,2 is not inA_3, ..., 9 ia not in A_{10} and 10,...,19 are not in A_1.
SO the intersection of the sets in A is empty.