Northeastern University
College of Engineering
Department of Electrical & Computer Engineering

EECE7205: Fundamentals of Computer Engineering

Spring 2022 – First Midterm Exam Review Questions

For questions **Q1** to **Q15** choose the **best** answer. Make a circle around your letter choice.

Q1. In the shown INSERTION-SORT algorithm, to sort list A into a decreasing instead of an increasing order, the following change(s) is(are) needed:

a. Step 4: $i = j + 1$
b. Step 5: while $i > 0$ and $A[i] < key$
c. Step 7: $i = i + 1$
d. All of the above.

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
4      i = j - 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i - 1
8      A[i + 1] = key
```

Q2. For the INSERTION-SORT algorithm in the previous question. Assume list A has **ten** elements and it is already sorted. How many actual lines of code will be executed?

a. 45 b. 46
c. 70 d. 80

Q3. The linear search algorithm checks every element in a list sequentially until the desired element is found. If the algorithm is applied on a list of *n* elements, then its asymptotic worst and best running times are:

a. $\Theta(n^2)$ and $\Theta(n)$ b. $\Theta(2n)$ and $\Theta(n)$
c. $\Theta(n)$ and $\Theta(1)$ d. None of the above.

Q4. Assume $f(n) = 2^n$ and $g(n) = 2^{n/2}$, then the following is true:

a. $f(n) = \Theta(g(n))$ b. $f(n) = O(g(n))$
c. $f(n) = \Omega(g(n))$ d. All of the above.

Q5. What are the minimum and maximum numbers of elements in a heap of height h?

a. 2^h and $2^{h+1} - 1$ b. $2^h - 1$ and 2^{h+1}
c. 2^{h-1} and $2^h - 1$ d. None of the above.

Q6. Is the heap represented by an array with values [23, 17, 14, 6, 13, 10, 1, 5, 7, 12] a max-heap?

a. Yes b. No

Q7. What are the leaves of the heap represented by an array with values:

[23, 17, 14, 6, 13, 10, 1, 5, 7, 12]?

a. [5, 7, 12] b. [7, 12]
c. [1, 5, 7, 12] d. [10, 1, 5, 7, 12]

Q8. A heap is represented by an array with values A = [27, 17, 3, 16, 13, 10, 1, 5, 7, 12, 4, 8, 9, 0]. The following will be the contents of A after applying the operation of MAX-HEAPIFY(A, 3).

a. [27, 17, 10, 16, 13, 9, 1, 5, 7, 12, 4, 8, 3, 0]
b. [27, 17, 10, 16, 13, 3, 1, 5, 7, 12, 4, 8, 9, 0]
c. [27, 17, 16, 3, 13, 10, 1, 5, 7, 12, 4, 8, 9, 0]
d. None of the above

Q9. What is the running time of the HEAPSORT algorithm on an array A of length n that is already sorted?

a. $\Theta(n)$ b. $\Theta(n \lg n)$
c. $\Theta(\lg n)$ d. None of the above.

Q10. What is the running time of the HEAPSORT algorithm on an array A of length n that is initially in reverse sorted order?

a. $\Theta(n)$ b. $\Theta(n \lg n)$
c. $\Theta(\lg n)$ d. None of the above.

Q11. Assume an initially empty stack S stored in array $S[1 .. 6]$ and the top of the stack is initially $S[1]$. What will be the content of S after the following operations in the sequence PUSH(S,4), PUSH(S, 1), PUSH(S, 3), POP(S), PUSH(S, 8), and POP(S)?

a. [4, 1, 3, 8] b. [8, 3, 1, 4] c. [4, 1] d. [3, 8]

Q12. Assume an initially empty queue Q stored in array $Q[1 .. 6]$ and both its head and tail are pointing to $Q[1]$. What will be the content of Q after the following operations in the sequence ENQUEUE(Q, 4), ENQUEUE(Q, 1), ENQUEUE(Q, 3), DEQUEUE(Q), ENQUEUE(Q, 8), and DEQUEUE(Q)?

a. [4, 1, 3, 8] b. [8, 3, 1, 4] c. [4, 1] d. [3, 8]

Q13. For hashing by the division method, which of the following is the best choice for the modulus m?

a. 256 b. 181 c. 128 d. 10

Q14. Given a hash table with size $\boldsymbol{m} = \mathbf{13}$ and starting index **0**, and a hash function $\boldsymbol{h(k) = k} \textbf{ mod } \boldsymbol{m}$. For $\boldsymbol{k} = \mathbf{25}$, state the hash position (home slot) and the following three positions if linear probing is used for collision resolution.

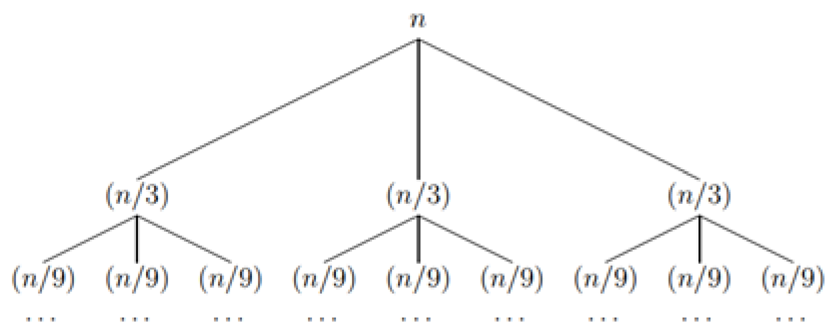
a. 12, 13, 14, 15 b. 11, 12, 13, 14
c. 12, 0, 1, 2 d. 11, 12, 0, 1

Q15. Suppose that we have numbers between 1 and 1000 in a binary search tree, and we want to search for the number 363. Which of the following sequences could **not** be the sequence of nodes examined?

```
a.   2, 252, 401, 398, 330, 344, 397, 363.
b. 924, 220, 911, 244, 898, 258, 362, 363.
c. 925, 202, 911, 240, 912, 245, 363.
d.   2, 399, 387, 219, 266, 382, 381, 278, 363.
e. 935, 278, 347, 621, 299, 392, 358, 363.
```

End of multiple-choice questions

Q16. What recurrence relation would generate the following recursion tree?



Q17. In order to sort $A[1..n]$ using a recursive version of the insertion sort algorithm, we recursively sort $A[1..n\text{-}1]$ and then insert $A[n]$ into the sorted array $A[1..n\text{-}1]$. Write a recurrence for the running time of this recursive version of insertion sort. Solve the recurrence to find the asymptotic notation of the running time.

Q18. Solve the following recurrence equations to find the big *O* asymptotic notation of the running time as a function of *n*:

a) T(n) = T(n-1) + n , and T(1) = 1

b) T(n) = T(n/2) + c, and T(1) = c (assume that $n = 2^k$ for a positive integer *k*)

Q19. Rewrite the shown ENQUEUE and DEQUEUE algorithms to detect underflow and overflow of a queue. Make any necessary assumptions.

```
ENQUEUE(Q, x)
Q[Q.tail] = x
if Q.tail == Q.length
    Q.tail = 1
else Q.tail = Q.tail + 1
```

```
DEQUEUE(Q)
x = Q[Q.head]
if Q.head == Q.length
    Q.head = 1
else Q.head = Q.head + 1
return x
```

Q20. Answer the following questions about the Binary Search trees:

a) Without drawing any trees, what are the minimum and maximum heights of binary trees with 63 nodes.

b) Starting from an empty binary search tree (BST), draw the tree after inserting the following values in the order given (from left to right).

`M, H, D, A, G, K, L, T, R, W, V, U`

c) On the same BST you created in part (b), add the following values B then C

d) Redraw the BST you have from part (c) after removing the following values:
G → K → M (in this order)

Q21. Answer the following questions about the Quicksort algorithm:

a) Illustrate the operation of PARTITION on the array
A = (13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11)

b) How would you modify QUICKSORT to sort into decreasing order?

```
QUICKSORT(A, p, r)
if p < r
    q = PARTITION(A, p, r)
    QUICKSORT(A, p, q - 1)
    QUICKSORT(A, q + 1, r)
```

```
PARTITION(A, p, r)
x = A[r]
i = p - 1
for j = p to r - 1
    if A[j] ≤ x
        i = i + 1
        exchange A[i] with A[j]
exchange A[i + 1] with A[r]
return i + 1
```

c) Can we replace the previous QUICKSORT pseudo code with the following TAIL-RECURSIVE pseudo code, which has one recursive call within a loop? Explain why.

```
TAIL-RECURSIVE-QUICKSORT(A, p, r)
while p < r
    // Partition and sort left subarray.
    q = PARTITION(A, p, r)
    TAIL-RECURSIVE-QUICKSORT(A, p, q - 1)
    p = q + 1
```

End of the Review Questions