Midterm-I **Comp Sci 2C03** - *Data structures and Algorithms* *Winter 2021*

Name _____ Student No._____

- For your exam you are ONLY allowed to refer to the lecture notes, text book, tutorials and assignments.
- Clearly write your full name and student number on each sheet.

Time allotted: 2 hours **Total Marks: 40**
(for the test and uploading it on Avenue)

Multiple choice questions [7 marks]

Circle your answer. Each question has just one correct answer. Therefore multiple selections will not get a mark.

1. **Suppose x is a linked-list node and not the last node/tail in the list. What is the effect of the following code fragment?** x.next = x.next.next;
   (a) Deletes from the list the node immediately following x.
   (b) Deletes from the list the node immediately preceding x.
   (c) Deletes from the list the node x.
   (d) None of the above.

2. **Suppose a stack implementation supports an operation `REVERSE`, which reverses the order of elements on the stack, in addition to the `PUSH` and `POP` operations. Which one of the following statements is `TRUE` with respect to this modified stack?**
   (a) A queue cannot be implemented using this stack.
   (b) A queue can be implemented where ENQUEUE will be implemented using a single operation and DEQUEUE will require two stack operations.
   (c) A queue can be implemented where ENQUEUE will be a combination of 3 stack operations and DEQUEUE will require a single operation.
   (d) A queue can be implemented where both ENQUEUE and DEQUEUE will require a single stack operation.

3. **The function $T(n) = \frac{n^3+2n+1}{n+1}$ is in**
   (a) $\Theta(n^2)$

(b) $O(n)$

(c) $\Omega(n^3)$

(d) None of the above.

4. **A binary tree has 63 internal nodes. What is the minimum height of this tree?**

   (a) 4

   (b) 6

   (c) 5

   (d) 1

5. **Suppose we are sorting an array A of 10 integers using top-down merge sort approach. During the execution the state of the array is $A =$ `1 2 5 3 7 9 8 10 12 11 14 19`. How many more merges are required to completely sort the entire array A?**

   (a) 3

   (b) 1

   (c) 2

   (d) 0

6. **A priority queue can be efficiently implemented using the queue data structure.**

   (a) True

   (b) False

7. **Heap sort is the only sorting algorithm with an efficient and easy in-place implementation, and which has an $n \log n$ worst case running time.**

   (a) True

   (b) False

Provide detailed answers to the 6 questions below

1. Write an implementation (either in pseudocode or JAVA format) of the operations `Push` and `Pop` for a Stack data structure (S) using a **double linked list**. You may assume that the other stack functions such as IsEmpty, Top, etc. are available to you. [5 marks]
   Answer: The solution follows the style of CLRS.

   procedure $Push(S, x)$
   if $!S.IsEmpty$ **then**

```
        x.next = S.head
        x.next.prev = x
    S.head = x
procedure Pop(S)
    if !S.IsEmpty then
        x = S.head
        S.head = S.head.next
        S.head.prev = NULL
        return x
    else
        return NULL
```

2. Consider the below program. Provide the running time function $T(n)$ for it, and explain your answer in few lines. Show that $T(n) \in \Theta(\log_2^2 n)$ using only the definition of big-Theta notation. [5 marks]

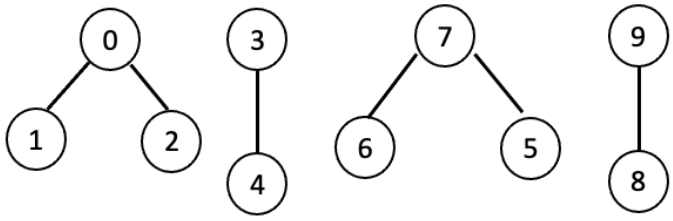
```
public class TwoSum
{
   public static int count(int[] a)
   {
      int N = a.length; Assume that N is a power of 2.
      int cnt = 0;
      for (int i = 1; i ≤ N; i = i+i)
         for (int j = 1; j ≤ N; j = j+j)
            cnt = a[i] + a[j];
      return cnt;
   }
   public static void main(String[] args)
   {
      int [] a = In.readInts(args[0]);
      StdOut.println(count(a));
   }
}
```

Answer: The program has two `for` loops. Since N is a power of 2, for each execution of the outer `for` loop, the inner `for` loop executes ~~$\log_2 n$~~ $(1+\log_2 n)$ times. Furthermore, since the outer `for` loop also executes ~~$\log_2 n$~~ $1+\log_2 n$ times, the running time of the algorithm is $T(n) = (1+\log_2 n)^2$. For ~~$c_1 = 1$, $c_2 = 0.5$ and $n_0 = 4$~~ $c_1 = 6$, $c_2 = 1$ and $n_0 = 4$, we have $c_2 \log_2^2 n \leq T(n) \leq c_1 \log_2^2 n$. Therefore $T(n) \in \Theta(\log_2^2 n)$.

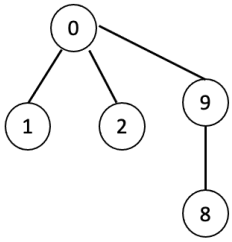
3. Assume the Weighted Quick-Union with Path Compression model and the below id[] array:

i	0	1	2	3	4	5	6	7	8	9
$id[]$	0	0	0	3	3	7	7	7	9	9

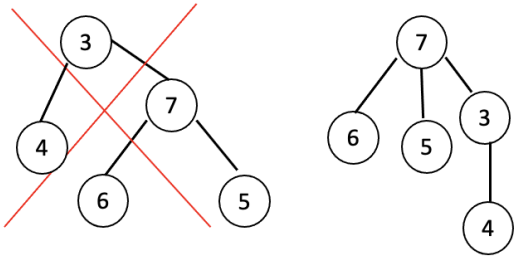
(a) Draw the trees represented by the above table. [2 marks]
Answer:



(b) Execute `union(1,8)` and draw the resulting tree. [2 marks]
Answer:



(c) Execute `union(4,5)` and draw the resulting tree.[2 marks]
Answer:



4. Give a double linked list implementation of Insertion sort algorithm using fewest insertions and/or data movement. You may assume that you have all the data values and operations supporting a double linked list L such as $L.head, L.tail$, and all the node (x) values and operations such as $x.next, x.prev, x.value$. Also, you may assume that all data values are positive integers and the input list contains at least two nodes. [6 marks]

```
procedure Insertion_Sort(L)
    x = L.head.next
    while x ≠ NULL do
        y = x.prev
        while y ≠ NULL and x.value < y.value do
            y = y.prev
        Insert_Node(L, x, y)
        x = x.next
    return L
procedure Insert_Node(L, x, y)
    if y ≠ NULL and x.value > y.value and y.next ≠ x then
        if x.next ≠ NULL then
            x.prev.next = x.next; x.next.prev = x.prev
        else
            x.prev.next = NULL
        x.next = y.next; x.prev = y
        y.next.prev = x; y.next = x
    if y = NULL then
        if x.next ≠ NULL then
            x.prev.next = x.next; x.next.prev = x.prev
        else
            x.prev.next = NULL
        x.next = L.head; x.prev = NULL; L.head = x;
```

5. Let `A= [2 4 5 7 1 2 10 11 3 2]` be the input array of size 10. As can be observed this array has 8 distinct numbers and 3 duplicates. Now consider the quicksort algorithm presented in Algorithm 2.5 (page 289 of textbook and slide# 20 in C2P2.pdf), and suppose that the initial random shuffle is omitted.

   (a) Give a permutation of A that requires least number of compares by this modified quicksort algorithm. Explain your answer in few lines. [3 marks]
   Answer: `A= [4 2 1 3 2 2 7 11 10 5]` - will require the least number of compares. As the pivot (first element of the input array) chosen will divide the input array into approximately equal size sub arrays, resulting in the $n \log n$ compares.

   (b) Give a permutation of A that requires maximum number of compares by this modified quicksort algorithm. Explain your answer in few lines. [2 marks]
   Answer: `A= [1 2 2 2 3 4 5 7 10 11]` - Any sorted (in increasing or decreasing order) version of A will result in maximum number of compares; that is, $O(n^2)$ compares.

6. (a) A Minimum Priority Queue is a collection that supports inserting an element and deleting the minimum element from the collection. Point out the changes in the implementation of the helper functions `swim` and `sink` presented on slides 8 and 10 of C2P3.pdf to support the `insert` and `delMin` operations. Comment

on whether you need to modify the existing `insert` and `delMax` operations for the Max.Priority Queue discussed in the textbook. If so, provide the changes. [3 marks]

Answer: The *less* function would be replaced by *greater* function in the sink and swim implementations for Max. Priority queue. The implementation for `insert` and `delMin` is same as the code for `insert` and `delMax`. Except that ideally the variable max in `delMax` would be renamed to min in `delMin`.

(b) How would you use the minimum priority queue to perform heapsort? For this question explaining in a few lines will suffice. No need to give implementation/code. [3 marks]

Answer:

(a) View input array as a complete binary tree.

(b) Heap construction: build a min-heap with all n keys.

(c) Sortdown: repeatedly remove the minimum key.

(d) The resulting array will be sorted in decreasing order. To sort in increasing order. exchange keys pairs $(1, n)$, $(2, n-1)$ and so on.