University of Nevada, Las Vegas Computer Science 477/677 Fall 2022

Assignment 3: Due Friday September 23, 2022, Midnight

Name:____

You are permitted to work in groups, get help from others, read books, and use the internet. Please follow Mr. Wang's instructions on how to submit your completed assignment.

1. Fill in the blanks. One word per blank.

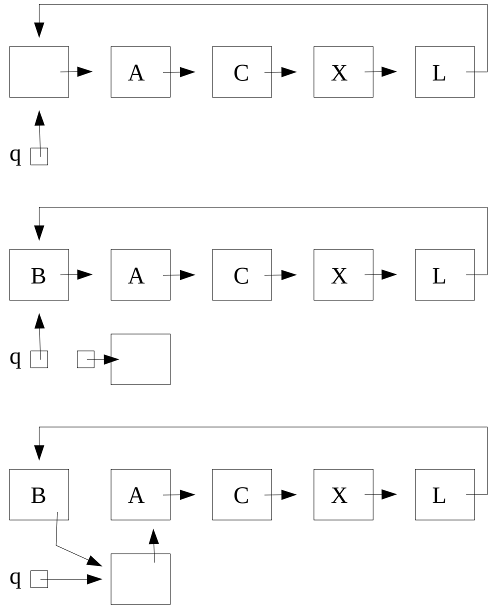
   (a) The worst case number of comparisons during an execution of a comparison based sorting algorithm with an input of size n is $\Omega((n \log n)$.

   (b) The items stored in a priority queue represent **unfulfilled obligations.**

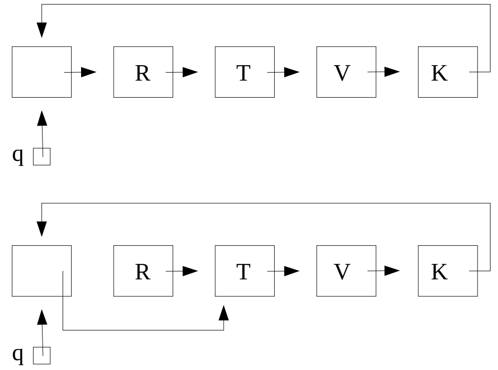
   A linked list has a *head* and a *rear*. Which of those will be the top, if linked list implements a stack? ________________

2. Explain how to implement a queue using a circular linked list. There should be only one pointer to the structure, to the dummy node.

   (a) Show the queue with the items A, C, X, L, in that order, from front to rear, then show how the queue changes when you insert B.



(b) Show the queue with the item R, T, V, K, in that order, from front to rear, then show how the queue changes when you execute dequeue.



Consider the function f(n) computed by the code below.

3.
```
int f(int n)
{
 // input condition: n >= 0
 if(n<7) return 1;
 else return f(n/2)+f(n/2+1)+f(n/2+2)+f(n/2+3)+n*n;
}
```

The function can be computed by recursion, as given in the C++ code above. However, we could also compute f(n) using dynamic programming or memoization.

(a) What is the asymptotic value of f(n)? The value itself, not the time to compute it. Write the recurrence, then solve it.

$f(n) = 4f(n/2) + \Theta(n^2)$
$f(n) = \Theta(n^2 \log n)$

(b) What is the asymptotic time complexity of the recursive computation of f(n)? Write the recurrence, then solve it.

$T(n) = 4T(n/2) + \Theta(1)$
$T(n) = \Theta(n^2)$

(c) What is the asymptotic time complexity of the dynamic programming computation of f(n)?

Compute $f(i)$ for all $i \leq n$ in order. The time complexity is $\Theta(n)$.

(d) What is the asymptotic time complexity of the computation of f (n) using memoization?

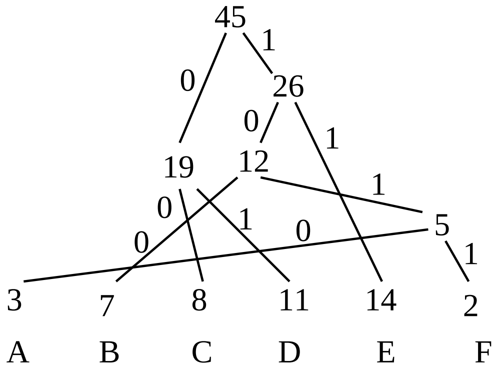
You will need to compute $f(m)$ for M distinct values of $m \leq n$, but M is much smaller than n. The time complexity is $\Theta(M)$.

To visualize the number of denominators needed, start with a large number, such as $n = 1000$. The denominators needed are
$1000,$
$500, 501, 502, 503,$
$250, 251, 252, 253, 254,$
$125, 126, 127, 128, 129, 130$
$62, 63, 64, 65, 66, 67, 68,$
$31, 32, 33, 34, 35, 36, 37,$
$15, 16, 17, 18, 19, 20, 21,$
$7, 8, 9, 10, 11, 12, 13.$

There are $\Theta(logn)$ blocks, each of which has no more than 7 values of m. Thus $M = \Theta(\log n)$.

4. Find an optimal Huffman code on the alphabet A,B,C,D,E,F with the following frequencies.

A	8	1010
B	7	100
C	3	00
D	11	01
E	14	11
F	2	1011


45
1
0
26
0
1
19
12
1
0
1
0
0
5
1
3
7
8
11
14
2
A
B
C
D
E
F