CSC 225 SPRING 2021 A01 & A02 (CRN 20713 & 20714)
ALGORITHMS AND DATA STRUCTURES I
FINAL EXAMINATION
UNIVERSITY OF VICTORIA

Student ID: ______________________________

Name: ___________________________________

DATE: 17 APRIL 2021
DURATION: 3 HOURS
INSTRUCTOR: RICH LITTLE

THIS QUESTION PAPER HAS **NINE** PAGES INCLUDING THE COVER PAGE.

THIS QUESTION PAPER HAS **EIGHT** QUESTIONS.

ALL ANSWERS CAN BE WRITTEN ON THIS EXAMINATION PAPER, ON YOUR OWN PAPER OR TYPESET, PROVIDED YOU SUBMIT ONE PDF DOCUMENT CONTAINING THEM.

THIS IS AN OPEN BOOK EXAM. YOU MAY REFERENCE ANY COURSE MATERIALS; SLIDES, TEXTBOOKS, CONNEX.

NO OTHER ACCESS TO THE INTERNET IS ALLOWED, WITH THE POSSIBLE EXCEPTION OF AN ONLINE PDF APPLICATION.

NO COMMUNICATION/COLLABORATION ALLOWED WITH ANY OTHER PERSON.

YOUR SOLUTIONS MUST BE SUBMITTED IN THE ASSIGNMENTS TOOL IN CONNEX BY 12 AM VICTORIA TIME. EXAMS SUBMITTED IN ANY OTHER WAY WILL NOT BE ACCEPTED UNDER ANY CIRCUMSTANCES. LEAVE YOURSELF PLENTY OF TIME TO DO THE SUBMISSION PROCESS.

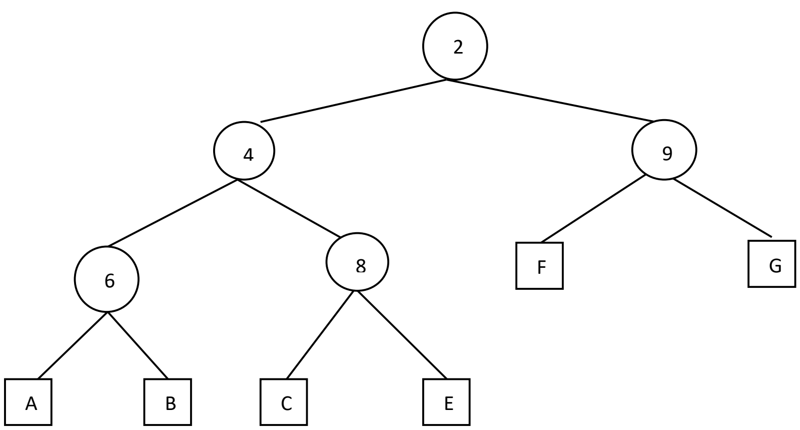
Q1 (8)	
Q2 (8)	
Q3 (8)	
Q4 (8)	
Q5 (8)	
Q6 (8)	
Q7 (8)	
Q8 (8)	
TOTAL (64) =	

1. (a). [3 marks] In terms of n, what is the total running time, $T(n)$, of this algorithm? Count assignments of s only.

```
Algorithm Loop(n):
    s ← 0
    for i ← 1 to n do
        for j ← 1 to 2n do
            s ← s + i
```

(b). [3 marks] How many arrangements of the letters in MISSISSIPPI have all four S's together?

(c). [2 marks] Consider the following heap with integer key values.



Which leaf would an item with key 1 be initially inserted into?

2. (a). [3 marks] Prove that $\log^2 n$ is $o(n^{1/2})$.

(b). [5 marks] Give a loop invariants argument to prove the correctness of selection sort.

```
Algorithm selectionSort(A,n):
    Input: Array A of size n
    Output: Array A sorted

    for k ← 0 to n–2 do
        min ← k
        for j ← k+1 to n–1 do
            if A[j] < A[min] then
                min ← j
        swap(A[k], A[min])
```

3. (a). [2 marks] Add up the number of nodes in a forest of at least two trees together with the number of edges. Can the number be even? Justify.

(b). [3 marks] What is the least number of internal nodes in a red-black tree with 3 red edges?

(c). [3 marks] What is the maximum number of nodes in a 2-3 tree with height 4 (recall, height of a leaf is 0 and leaves are empty/null)?

4. Let T be a proper binary tree. Let the height of the tree be denoted by h. Justify all your answers below.

(a). [1 marks] What is the minimum number of external nodes in T in terms of h?

(b). [1 marks] What is the maximum number of external nodes in T, also in terms of h?

(c). [1 marks] What is the minimum number of internal nodes in terms of h?

(d). [1 marks] What is the maximum number of internal nodes in terms of h?

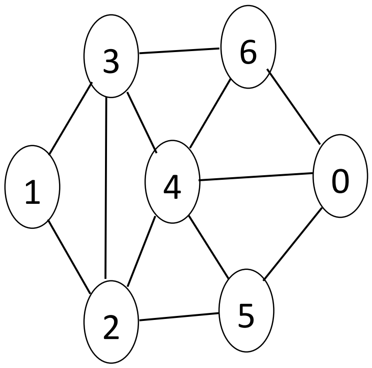
(e). [2 marks] Let n be the total number of nodes in T. What are the lower and upper bounds on n, in terms of height h?

(f). [2 marks] What are the lower and upper bounds on the height of the tree in terms of n?

5. (a). [6 marks] Insert the keys R E D S O X into an initially empty left-leaning red-black tree, in the given order. Show your work, there should be at least six trees along the way. Any time you need to invoke a `rotateRight(h)`, `rotateLeft(h)` or `flipColors(h)`, indicate it between trees and specify the node `h` to be passed to the method by its key.

(b). [2 marks] Draw the 2-3 tree that corresponds to the final red-black tree in (a).

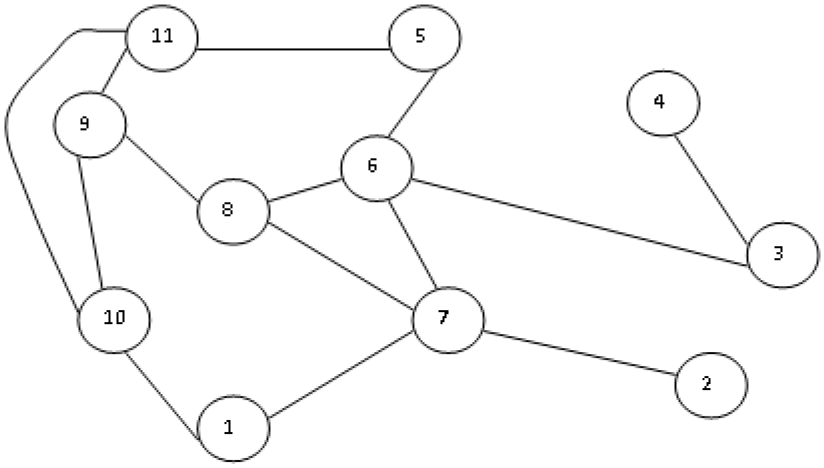
6. (a). [3 marks] Consider the following simple undirected graph G and draw the adjacency-matrix representation of it.



(b). [3 marks] Using pseudocode, write the graph method `numEdges()` which takes as input a simple undirected graph G, in adjacency-matrix form, and an integer n, the number of vertices in G (i.e. the size of the matrix). Your method should return an integer containing the number of edges in the graph.

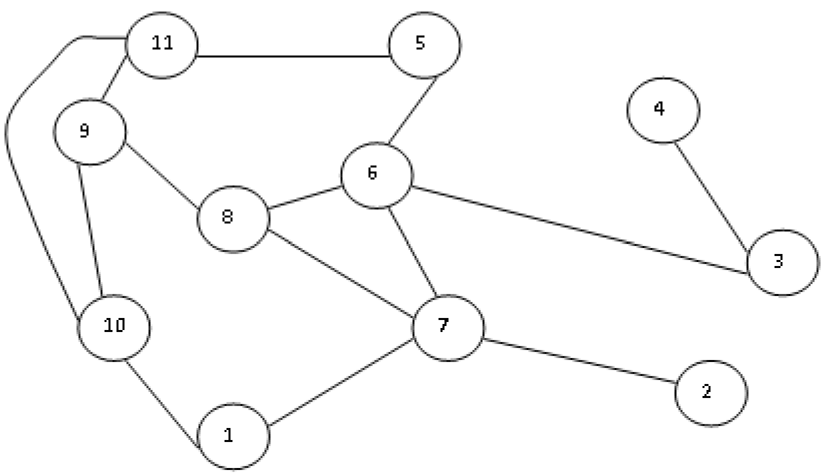
(c). [2 marks] Analyze the worst-case running time of your algorithm in (b).

7. (a). [3 marks] Consider the following undirected graph *G*. Order the vertices as they are visited in a BFS traversal, starting at vertex 1. Assume adjacent vertices are given in ascending order according to their label.



(b). [1 mark] On the graph, indicate which edges are *discovery* edges and which are *cross* edges.

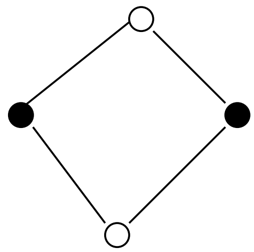
(c). [3 marks] Now consider the same graph *G*, again assuming adjacent vertices are given in ascending order according to their label. This time order the vertices according to their postorder, using a DFS traversal.



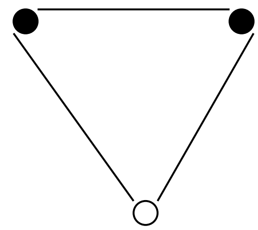
(b). [1 marks] On the graph, indicate which edges are *discovery* edges and which are *back* edges.

8. [8 marks] A *2-colouring* of an undirected graph with n vertices and m edges is the assignment of one of two colours (say, black or white) to each vertex of the graph, so that no two adjacent nodes have the same colour. So, if there is an edge (u,v) in the graph, either node u is black and v is white or vice versa. Give (pseudocode!) an $O(n + m)$ time algorithm to 2-colour a graph or determine that no such colouring exists, and justify the running time. The following shows examples of graphs that are and are not 2-colourable:

2-colourable



Not 2-colourable



THE END