University of Manitoba
COMP 2140, Winter 2019
Assignment 5

Due Date: April 9, at 10:00pm

'The two most important days in your life are the day you are born and the day you find out why ...'
Mark Twain

Please pay attentions to the followings when preparing your assignment:

• There is only written component in this assignment. You need to submit your solutions via Crowdmark.

• There are 5 problems. All questions are mandatory. Some of the problems related to the graph theory are not covered in the lectures yet. We hopefully cover all of them by Friday. Note that all materials for approaching these questions are added to the course webpage and I encourage you to go through them before we cover them. Graph problems are designed in a way to take less time from you.

• As always, I encourage you to ask your questions on Piazza. Those students who help others on Piazza will receive bonus marks (by ‘helping’, I mean removing confusions). It is likely that I drop hints on Piazza in response to the questions that are asked publicly. In the pursuit of fairness, I hesitate to drop a hint in response to private questions or emails. So, be active on Piazza!

• This is an individual assignment. You are welcome to discuss questions with your friends (or enemies). But you should write the answers individually, and you should fully understand what you are writing. Please read http://www.cs.umanitoba.ca/~kamalis/winter19/infoCOMP2140.pdf for guidelines on academic integrity.

• Good luck!
Problem 1  B-Trees

Consider the following B-tree of min-size 2. Note that in such tree, each node should have at least 2 and at most 4 keys.

(a) Insert the following keys (in the same order) into the tree: 65, 60, and 25.
You just need to draw one tree (the final tree after all insertions).

Answer: Insertion of 60 and 65 does not cause an overflow. Insertion of 25 causes an overflow; as a result, its node is divided into two nodes and the middle item (among the five in the overflowed node) is inserted to the parent (24 is recursively inserted above which causes another overflow, and 24 is yet again moved forward to the root).

(b) From the original tree (the above tree), delete the following keys (in the same order): 55, 16, and 57.

Answer: Deletion of 55 does not cause an underflow. Deletion of 16 causes an underflow; fortunately one of the direct siblings (the one on the right) has more than minimum keys; hence it borrows a key (21) to the parent, and the parent borrows a key (20) to resolve underflow. Deletion of 57 cause an underflow and this time the direct sibling cannot help. We merge the underflow node with the direct sibling and borrow a key (50 from the parent). That causes anotehr underflow in the parent which is resolved by borrowing from a direct sibling.
Problem 2  Heap Sort

In order to sort a set of \( n \) items, we can insert them one by one into an initially empty heap (using the `insert` operation) and then apply the `extractMax` operation \( n \) times. This way, items are extracted in the reversely sorted order.

(a) Write down the time complexity of Heap-Sort using big-O notation. Provide a short justification of your answer.  
**Answer:** Each insert and extract-Max operation takes \( O(\log n) \). The first \( n \) insert operations take \( O(n \log n) \) and it is followed by another \( O(n \log n) \) for extracting the maximum item (among the remaining ones) \( n \) times. So, heap-sort takes \( O(n \log n) \).

(b) Indicate whether the **asymptotic time complexity** (which is denoted in big-O notation) improves if the first \( n \) insertions are replaced by a makeHeap (heapify) operation. Recall that makeHeap runs in \( O(n) \). Provide a short justification for your answer.  
**Answer:** The time complexity does not change because the second phase (extract-Max operations) still take \( O(n \log n) \). We somehow improve from \( O(n \log n) + O(n \log n) \) to \( O(n) + O(n \log n) \). Both these functions are indeed \( O(n \log n) \).

Problem 3  Heap Operations

(a) Consider a heap stored in an array \( a = [20, 12, 18, 9, 3, 16, 17, 2] \). Write the updated array when we apply `extractMax` operation.

(b) Consider the same heap \( a = [20, 12, 18, 9, 3, 16, 17, 2] \). Write the updated array when we apply the operation `insert(19)`.

(c) Indicate whether the following statement is correct or not. Provide a brief justification for your answer. You should either use an example (with large \( n \)) to show the statement is not correct or use English words to prove the statement:  
"The `extractMax()` operation in a heap of size \( n \), where \( n \) is a large number, can take \( O(1) \) in the best case".  
**Answer:** See the following figures. Note that for part (c), `extractMax` involves placing 500 in the root, swapping it with 1000, and stopping.

Problem 4  Spanning Trees

Consider the following graph:
(a) write down the first 5 edges that are selected by the Kruskal’s algorithm. You just need to provide a list of edges (indicate each edge with its endpoints, e.g., (a,b)).

(b) write down the first 5 edges that are selected by the Prim’s algorithm. You just need to provide a list of edges (indicate each edge with its endpoints, e.g., (a,b)).

Answer:  
Kruskal: (c, f), (d, f), (h, i), (e, h), (e, g)  
Prim: (c, f), (d, f), (f, g), (e, g), (e, h)

Problem 5  Planar Graphs & Colouring

Consider the following graph:

(a) The graph is partially colored with green (G), red (R), and blue (B) colors. Complete the coloring using these three colors. If you think the graph is not colorable using 3 colors, just indicate so; no need to justify.

(b) Show the above graph is planar by providing a planar embedding of it. If you think the graph is not planar, just indicate so; no need to justify.

Answer:  The graph is both 3colorable & planar.