Student name:
Student number:

COMP SCI 2C03

Term: April 2020
Instructor: Dr. R. Janicki
Duration: 4.0 h (Exam only 3.0 hrs, extra time for using technology: 1hr)
McMaster University Final Examination

Take home examination

THIS EXAMINATION PAPER INCLUDES 5 PAGES AND 18 QUESTIONS. It consists of 15 standard questions and 3 more theoretical questions.
TOTAL = 192 pts (standard: 162, theory: 30)

Officially the exam starts at 12:30pm and ends at 3:30pm (for students without extra time permissions)

Technically, the exam was posted on the course website and sent to you via e-mail at 12:15pm.

You should submit it back via SVN as PDF file and, just in case, send it to Ryszard Janicki at janicki@mcmaster.ca via standard e-mail with an attachment, BEFORE OR ON 4:15 pm. SVN will be closed at 4:20pm and I will use e-mail time stamp for a verification (if needed).

For questions use the slack channel, or if necessary send an e-mail directly to Ryszard Janicki at janicki@mcmaster.ca, if the last digit of your student number is even (2,4,6,8,0), or to Morteza Alipour Langouri at alipoum@mcmaster.ca, if the last digit of your student number is odd (1,3,5,7,9).

STANDARD QUESTIONS: 162 pts

1.[5] Consider the following sequence of pairs p-q: 0-1, 3-4, 2-3, 1-5, 4-5. Show the contents of the id[] array and the number of times the array is accessed for each input pair when you use weighted quick-union *with path compression*. Use path compression each time when it is needed.

2.[5] Consider Quick Sort algorithm with the following procedure for finding pivot (splitter). The function *pivot* scan the array from the smallest index until it finds two different elements. The larger element becomes the pivot (splitter). If all elements are the same no sort is needed. Give an example of a sequence (at least 8 elements long) for which this version of Quick Sort needs O(n^2) steps. Prove that your example is correct.

3.[5] Give traces, in the style of Algorithm 2.4 and Assignment 1, showing how the keys 4, 3, 1, 9, 6, 2, 7, 0, 6 are sorted with *top-down mergesort*.

4.[19] a.[5] Show the Binary Search Tree that results if the integers 4, 3, 1, 9, 6, 2, 7 are inserted into an empty tree. What is the result of deleting 4? Show all the steps.

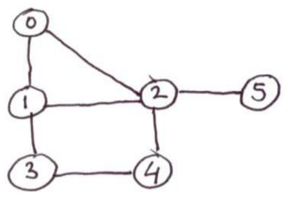
b.[7] Show the 2-3 that results if the integers 4, 3, 1, 9, 6, 2, 7 are inserted into an empty tree. Show all the steps.

c.[7] Show the Red-Black tree that results if the integers 4, 3, 1, 9, 6, 2, 7 are inserted into an empty tree. Show all the steps.

5.[4] Suppose that the sequence 4, 3, *, 1, 9, 6, * (where a number means *insert* and an asterisk means *remove the maximum*) is applied to an initially empty priority queue represented a max-oriented heap. Give the sequence of heaps produced by operations described by this sequence.

6.[8] Sort the sequence 4, 3, 1, 9, 6 using (min-oriented) heap sort.

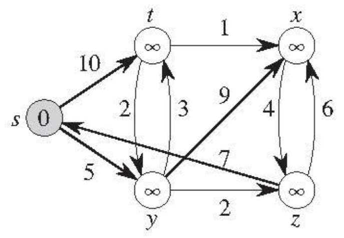
7.[6] Consider the graph G below:



a.[3] Find a Depth-First Search tree *T* for the above graph starting with the vertex 0. Show all the vertices as they are discovered in sequence starting from 1 to the last vertex included in *T*.

b.[3] Find a Breadth-First Search tree *T* for the above graph starting with the vertex 0. Show all the vertices as they are discovered in sequence starting from 1 to the last vertex included in *T*.

8.[37] Consider the following directed graph *G*.



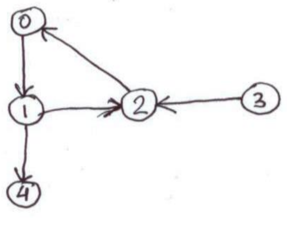
a.[5] Draw a representation of G by adjacency list assuming the *alphabetic* ordering of each list.

b.[17] Use the Dijkstra's algorithm to find the shortest paths from *s* to the other vertices. Assume that the order of vertices on the list of adjacent vertices is the alphabetical order. Also construct the P array (recovering the paths) step by step. Recover all paths. Assume the list of adjacent vertices implementation and with alphabetic ordering of each list. Use heap to implement the set V– S. Show all steps.

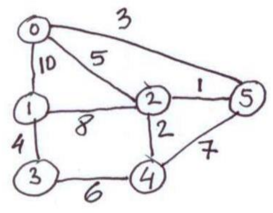
c.[15] Use the Bellman-Ford algorithm to find the shortest paths from *s* to the other vertices. Show all steps. Assume that the order of vertices on the list of adjacent vertices is the alphabetical order.

9.[10] Consider inserting the keys 28, 19, 8, 23, 34, 12, 17 into a hash table of length *m* = 11 using open addressing with auxiliary has function *h'*(*k*) = *k* mod m. Illustrate the result of inserting these keys using *linear probing.*

10.[13] Consider the connected directed graph below. Find all its strongly connected components. Use Kosaraju-Sharir algorithm. Illustrate all steps.



11.[12] Consider the graph below.



a.[6] Find Minimum Spanning Tree using Kruskal's Algorithm. Show all steps.

b.[6] Find Minimum Spanning Tree using Prim's Algorithm. Show all steps.

12.[10] a.[4] Give a trace for LSD string sort for the keys:
no th fo go to tr ai ta pa

b.[6] Give a trace for MSD string sort for the keys:
never do this in my house
are inserted in that order into an initially empty trie (do not draw null links).

13.[13] Consider the pattern P = aab and the string
S = abbbababaab
Compute the failure function for P, and then use Knuth-Morris-Pratt algorithm, as described in Lecture Notes 18, to verify if P appears in S. Show all steps. How many character comparisons are required by the KMP algorithm?

14.[10] Consider the pattern P = aab and the string
S = abbbababaab
Use Boyer-Moore algorithm to verify if P appears in S. Show all steps. How many character comparisons are required by the Boyer-Moore algorithm?

15.[5] Let a, b, c, d, and f be six characters with probabilities 0.4, 0.17, 0.15, 0.13, 0.1 and 0.05 respectively. Construct an optimal Huffman code and compute the average code length.

THEORY QUESTIONS: 30pts

16.[7] The standard insertion sort algorithm has both the worst case and average case time complexities $\Theta(n^2)$. Make a modification to this algorithm so that the worst case time complexity is $O(n \log n)$.

17.[15] Write an efficient algorithm to find the minimum number of edges that need to be removed from an undirected graph $G=(V,E)$ so that the resulting graph is acyclic. What is the time complexity of your algorithm? Assume the graph is represented by adjacency lists.
(*Hint*: Find the connected components of G first.)

18.[8] Write an $O(n)$ time algorithm to determine if an undirected graph is actually a *tree* (i.e. it is acyclic and connected). Assume that the graph is represented as adjacency list.

END OF EXAM