Name: ____________________ **NSID:** _______________ **Student #:** __________

University of Saskatchewan
Department of Computer Science
CMPT 360 - Fall 2019
Instructor: Debajyoti Mondal
Midterm Examination
October 28, 2019

Marks: 40 **Time: 40 minutes**

Instructions:

- Write your name, NSID, and student number.
- Hand in every page, even if you rip it off the booklet.
- Read every question carefully.
- Write the answers in the space provided. Use the back page if you need extra space (or for rough work).
- The mark value of each question is provided in the left margin.
- No textbook. No calculators. No notes. No laptops. No phones.
- If you have a clarifying question, please raise your hand and I or the TA will come to you. I will only answer clarification questions. No hints.

Academic Honesty: This exam is an individual undertaking – cheating on an exam is considered a serious offence by the University and can be met with disciplinary action, including suspension or expulsion. By handing in this exam you affirm that this work is entirely of your own.

Please do not write in the table to the right. For marking purposes only.

Page	Points	Score
1	20	
3	10	
4	10	
Total:	40	

True/False

Write T or F in the box depending on whether the statement is True or False, respectively.

(2) 1. $\boxed{}$ The height of every Huffman tree with n keys is $O(\log n)$.

Answer: FALSE (the tree is full not necessarily balanced)

(2) 2. $\boxed{}$ Any algorithm that computes the median of an n element array must require at least $\Omega(n \log n)$ time.

Answer: FALSE (The class covered a linear time algorithm)

(2) 3. $\boxed{}$ By master theorem, $T(n) = 16T(n/4) + n^2$ is upper bounded by $O(n^2 \log n)$.

Answer: TRUE (Master Theorem)

(2) 4. $\boxed{}$ Subset sum can be solved in polynomial time, i.e., $O(n^c)$ time where c is a constant.

Answer: FALSE (The best known algorithms are either exponential or pseudo-polynomial - both covered in the class and book.)

(2) 5. $\boxed{}$ A matching between n jobs and n employees is stable if it contains at least n stable pairs.

Answer: TRUE (In the setting of the book's stable matching problem, every matching will contain exactly n stable pairs.)

(2) 6. $\boxed{}$ The time complexity of the recurrence $T(n) = T(n/5) + T(n/4) + O(1)$ is bounded by $O(n)$.

Answer. TRUE (Guess and verify.)

(2) 7. $\boxed{}$ A topological sorting of an n vertex graph takes $O(n)$ time.

Answer. FALSE (The edges should be taken into consideration.)

(2) 8. $\boxed{}$ The Greedy schedule algorithm will also give optimal result if we sort the classes based on starting time and greedily choosing the classes that started late.

Answer. TRUE (Earliest finish time is equivalent to the most delayed starting time)

(2) 9. $\boxed{}$ A Huffman code is a prefix-free binary code.

Answer. TRUE (Covered in the class, also see the book)

(2) 10. $\boxed{}$ A divide and conquer algorithm is a simpler form of dynamic programming.

Answer. FALSE (The substructure property that allows reuse of the subproblems may be missing in a divide and conquer algorithm - however, everyone was graded with full marks for this question.)

[Optional] You may want to use the following space to explain your true/false label for a question that you found difficult to answer. For example, you thought that the statement could be interpreted both as a true and a false statement.

Your explanation may not change your mark on that question. I encourage first to finish writing your exam and then come back here later (if you have time left to revise).

Short Questions

(5) 11. How many strongly connected components can there be in a directed acyclic graph with n nodes? Give a brief argument for your answer.

Answer. Exactly n components.

(Proof by contradiction): Suppose for a contradiction that there is less than n strongly connected components. Then there is a strongly connected component C with at least two nodes. Since any two nodes in C are reachable from each other, it contains a cycle, a contradiction that the original graph is a DAG.

(5) 12. Draw the Huffman tree for the keys k_1, k_2, k_3, k_4, k_5, where the frequencies of the keys are $8, 4, 4, 2, 2$, respectively.

See book example. Section 4.4 Huffman Codes Page 167.

(5) 13. Suppose you are given an array $A[1, n]$ of n numbers, which may be positive, negative, or zero. Write down an efficient algorithm that would return true if it finds a subarray $A[i, j]$ that sum to 0. For example, $A = [-1, 2, -3, 1, -2]$ has a subarray $A[2, 4]$ that sum to 0. What is the complexity of your algorithm?

See Section 3.9 Optimal Binary Search Trees Page 118. $O(n^2)$-time algorithm.

(5) 14. Write down the recurrence relation for computing a maximum independent set on a binary tree. What is the running time of your algorithm?

See Section 3.10 Dynamic Programming on Trees. $O(n)$-time algorithm.