Computer Science C73
Scarborough Campus

September 16, 2019
University of Toronto

Homework Assignment #2
(worth 6% of the course grade)
Due: September 23, 2019, by 10 am

- *You must submit your assignment as a PDF file through the MarkUs system by logging in with your UTORid at* `markus.utsc.utoronto.ca/cscc73f17`. *To work with a partner, you and your partner must form a group on MarkUs.*
- *It is your responsibility to ensure that the PDF file you submit is legible. To this end, I encourage you to learn and use the LaTex typesetting system, which is designed to produce high-quality documents that contain mathematical notation. You are not required to produce the PDF file you submit using LaTex; you may produce it any way you wish, as long as the resulting document is legible.*
- *By virtue of submitting this assignment you (and your partner, if you have one) acknowledge that you are aware of the policy on homework collaboration for this course.*[a]
- *For any question, you may use data structures and algorithms previously described in class, or in prerequisites of this course, without describing them. You may also use any result that we covered in class, or is in the assigned sections of the official course textbooks, by referring to it.*
- *Unless we explicitly state otherwise, you may describe algorithms in high-level pseudocode or point-form English, whichever leads to a clearer and simpler description.*
- *Unless we explicitly state otherwise, you should justify your answers. Your paper will be graded based on the correctness and efficiency of your answers, and the clarity, precision, and conciseness of your presentation.*

[a] "In each homework assignment you may collaborate with at most one other student who is currently taking CSCC73. If you collaborate with another student on an assignment, you and your partner must submit only one copy of your solution, with both of your names. The solution will be graded in the usual way and both partners will receive the same mark. Collaboration involving more than two students is not allowed. **For help with your homework you may consult only the instructor, TAs, your homework partner (if you have one), your textbook, and your class notes. You may not consult any other source.**"

Question 1. (15 marks) A summer camp has n campers, labeled $1, 2, \ldots, n$, and wants to organize a canoe trip. Camper i weighs w_i. Each canoe must have exactly two campers, whose combined weight must not exceed C; naturally, no camper can be in two canoes! Our job is to pair-up as many campers as possible subject to the weight constraint.

More precisely, a set of (unordered) pairs of campers is ***feasible*** if (a) no camper belongs to two different pairs in the set; and (b) for any pair $\{i, j\}$ in the set, $w_i + w_j \leq C$. A feasible set of pairs of campers is ***optimal*** if there is no feasible set with greater cardinality. Given $w_1, \ldots, w_n$ and C, we want to find an optimal set of pairs of campers.

a. Prove that the following greedy algorithm does not necessarily find an optimal set: If $n < 2$, return the empty set. Otherwise, let p and q be two lightest campers (ties broken arbitrarily). If $w_p + w_q \leq C$, return the set of pairs obtained by adding the pair $\{p, q\}$ to the set returned by recursively applying the algorithm to the remaining campers; otherwise, return the empty set.

b. Prove that the following greedy algorithm always finds an optimal set: If $n < 2$, return the empty set. Otherwise, let p be a lightest camper and q be a heaviest camper (ties broken arbitrarily). If $w_p + w_q \leq C$, return the set of pairs obtained by adding the pair $\{p, q\}$ to the set returned by recursively applying the algorithm to the remaining campers; otherwise, discard q and return the set returned by recursively applying the algorithm to the remaining campers. The recursion ends when the number of remaining

campers is less than two. (**Hint.** Prove that if the combined weight of a lightest and heaviest camper does not exceed C, there is an optimal set that contains a pair consisting of these two campers.)

c. Suppose the camp has only four-person canoes; i.e., each canoe must carry exactly four campers. Consider the following greedy algorithm for this case: Take the two lightest and two heaviest campers (ties broken arbitrarily). If the combined weight of these four campers is at most C add this group of four campers to the set returned by recursively applying the algorithm to the remaining campers; otherwise discard the heaviest of the four campers and return the set returned by recursively applying the algorithm to the remaining campers. The recursion ends when the number of remaining campers is less than four. Does this algorithm work? Justify your answer

Question 2. (10 marks) Consider Huffman's algorithm.

a. (2 marks) Give an example of a (small) set of symbols and their associated frequencies so that there is exactly one symbol with frequency 1/3, all other symbols have frequency (strictly) less than 1/3, and Huffman's algorithm may produce a codeword of length 1.

b. (8 marks) Prove that for any set of symbols, if every symbol has frequency (strictly) less than 1/3, Huffman's algorithm cannot produce a codeword of length 1.

Question 3. (10 marks)

a. (8 marks) Modify Dijkstra's algorithm to compute the ***number*** of minimum-weight paths from the start node s to every node, assuming that all edges have (strictly) positive weights. Your modification should have the same running time (asymptotically) as Dijkstra's algorithm. You need not prove that your algorithm is correct — but it should be! (To convince yourself that it does, I suggest that you revisit the invariants that we used in the proof correctness of Dijkstra's algorithm and that you consider how they should be strengthened to prove the correctness of your algorithm. You should not include this in your answer, however.)

b. (2 marks) Does your algorithm work if edges have non-negative (as opposed to strictly positive) weights — i.e., if some edges may have zero weight? Justify your answer.