Computer Science C73 — September 9, 2026
Scarborough Campus — University of Toronto

Homework Assignment #1
Due: September 23, 2026, by 11:59 pm

- ***You must submit your assignment through the Crowdmark system.*** *You will receive by email an invitation through which you can submit your work in the form of separate PDF docucments with your answers to each question of the assignment. To work with a partner, you and your partner must form a group on Crowdmark.* ***The group must be formed before either partner submits <u>any</u> answer.***
- *To minimize bias in grading, Crowdmark does not reveal your name to the grader. Please do not subvert this feature by including your name(s) on the files you submit.*
- *The PDF files you submit must be legible without undue effort by the grader. 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 files you submit using LaTex; you may produce them 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 course notes 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. Do not provide executable code.*
- *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] "For help with your homework you may consult only the instructor, TAs, your homework partner (if you have one), the CSCC73 Course Notes, and your personal class notes. You may not consult any other source, including AI-based tools."

Question 1. (20 marks) Let $\mathcal{I}$ be a sequence of closed intervals over the reals. A set of numbers C is a ***point cover*** of $\mathcal{I}$ if every interval in $\mathcal{I}$ contains at least one number in C (that is, for each $I = [s, f]$ in $\mathcal{I}$, there is some $c \in C$ such that $s \leq c \leq f$). C is a ***minimum point cover*** of $\mathcal{I}$ if it is a minimum cardinality point cover of $\mathcal{I}$.

a. Consider the following greedy algorithm.

```
1  C := ∅
2  while I ≠ ∅ do
3      let c be a number that is contained in the largest number of intervals in I (ties resolved arbitrarily)
4      C := C ∪ {c}
5      delete from I all intervals that contain c
6  return C
```

Prove that this algorithm does not always find a minimum cover of $\mathcal{I}$.

b. Describe an efficient greedy algorithm that given a sequence of intervals $\mathcal{I}$ (in arbitrary order) returns a minimum point cover of $\mathcal{I}$. Prove the correctness of your algorithm and anlyze its running time.

Question 2. (15 marks) Consider the following process: We are given a positive integer n. We start with $x = 1$. At each step of the process we can apply to x one of two operations: INCREMENT x by 1 (i.e., $x := x + 1$) or DOUBLE x (i.e., $x := 2x$). We want to reach the given number n (i.e., make $x = n$) by

applying repeatedly such steps. For example, if $n = 1024$, we can do this in 1023 increment steps, or in 10 double steps, or in some combination of these two kinds of steps.

Our problem is the following: Find a sequence of increment or double operations so that starting from 1 we reach n *with the fewest steps*. Describe a greedy algorithm that solves this problem. Your algorithm should take as input the positive integer n, and should produce as output a sequence each element of which contains I (increment) or D (double), so that if we apply these operations to x in the given order starting with $x = 1$, we end with $x = n$. Analyze the running time of your algorithm and prove its correctness.

Question 3. (25 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 4. (10 marks) Consider Huffman's algorithm.

a. Give an example of a (small) set of symbols and their associated frequencies so that the maximum frequency of any symbol is equal to 2/5, and Huffman's algorithm *could* produce a tree in which no codeword has length 1. Show such a tree that could be produced by Huffman's algorithm in your example.

b. Prove that for any set of symbols, if some symbol has frequency (strictly) greater than 2/5, Huffman's algorithm will necessarily produce a codeword of length 1.