General Instructions

The practice problems are divided according to the topics in the course. It is strongly recommended that you make sure you've mastered each topic in sequence before moving on to more advanced problems, since the problems build on each other.

If you find you need help on any of the problems, contact Mike Stachowsky (mstachowsky@uwaterloo.ca) or ask a member of the teaching team.

Problem Labelling

The problems are labelled as follows:

prog_drill – MN – title

M is a number relating to the order in which the problem sets should be attempted, with lower numbered problems to be done first.

N is a number indicating a perceived difficulty level. A level 0 problem is considered to be easier than a level 3 problem. Generally, an easier problem may involve fewer lines of code, fewer concepts from previous sections, and may be more intuitive to understand. It's important to realize that you may find some problems easier or harder than others. This is only indicative of your personal programming style, and the difficulty levels should be taken as suggestions only.

Testing

The problems have been provided with a list of input/output tests that can serve as a starting point for you to test your program. We have given you a set of inputs and the corresponding output of a working program. We have tried to make these tests comprehensive and to include important edge cases, but you should also try to come up with your own test cases.

Time Target

The time targets provided are expectations of the time it should take a programmer **in an exam environment.** There are three levels, indicated by one, two, or three stars. An average student who has prepared successfully should be earning at least two stars on every question that they attempt. If you find yourself consistently earning only one star, we suggest that you seek help from the teaching team as soon as possible.

Since your exams will be written on paper, you may choose to first write out your solution on paper, time yourself, and then copy the answer into your coding environment to see if it works. This way, you can practice writing code by hand and can check your answers as you go.