Homework 1

Please submit your work by Wednesday, April 9 at 11:59pm. You should submit one file `hw1/hw1.sml` in your subversion repository.

The purpose of this assignment is to allow you to gain experience programming in Standard ML. The particular set of functions you are asked to write is admittedly not *especially* relevant to the study of programming languages; however, familiarity with ML is essential moving forward.

The ML way of doing things is pervasive in the academic study of programming languages generally, and pervasive in our course materials (the textbook, for example) specifically.

Standard ML is a clean, compact, easy-to-understand, carefully designed programming language with a rigorous and helpful static checking system. David MacQueen, one of the original designers, said that to the greatest possible extent the language had had the "complexity engineered out of it" (a paraphrase). It is hard to overstate the influence of ML on the programming languages research community, even though ML itself is not in nearly as widespread direct use as some of us believe it deserves to be.

To complete homework 1, download this starter fileand commit it to a new directory `hw1` in your subversion repository.

Read carefully through the starter file. All the functions you must complete for the homework assignment are specified either by name and type only, which in many cases makes the intention of the function clear enough, or, where needed, by name and type with an explanatory comment. I expect that questions about any ambiguities will percolate up to piazza. As given in the starter file, all functions immediately raise an exception rather than compute anything (which is, in any case, the best, safest stand-in for function bodies not yet written).

Please note there is a lightweight checking infrastructure included in the starter file in the module `Check`. This is a return of sorts to`check-expect`, which I know alums of CS151 sorely miss. Read the implementations inside the `Check`module and it should be clear how to use them. Some skeletal calls to `Check` tests are sprinkled in the starter file, in comments. Please do not ask us how many tests are required! You are grownups now. Test each function sufficiently well to convince yourself that it works.