Exam Rules

- You have **110 minutes** to complete this exam. You must stop working once the instructor calls for papers. You may receive a deduction if you keep working after the instructor calls for papers.
- The exam is open-book/notes. You must work alone and may not use any computing devices including calculators. Cell phones, music players, and other electronics may NOT be out during the exam for any reason.
- Please be quiet during the exam. If you have a question or need, please raise your hand.
- Corrections or clarifications to the exam will be written at the front of the room.
- Please obey the University Code of Conduct during the exam.
- Calculators are not allowed during the exam. Cell phones, music players, and other electronics may NOT be out during the exam for any reason.

Programming Guidelines:

Unless otherwise noted, you may call standard library functions available in the top-level environment, such as:

- **ML**: operators (-, +, -, *., div, mod, ::, @, ^, o, not, andalso, orelse, <, >, <=, >=, =, <>), numeric functions (e.g. abs and Int.max), list functions (e.g. hd, tl, length, rev, foldl, foldr), conversion functions (real, trunc, floor, ceil, ord, chr, str), string functions (implode, explode, concat, size), standard tuple functions (#1, #2, etc.), and any other functions from ML basic data type structures such as Int, Real, String, Bool, and Char (but not List);
  - fun quicksort(f, list), fun m -- n,
  - fun mapx(f, list), fun map f list,
  - fun filterx(f, list), fun filter2 f list,
  - fun reduce(f, list), fun reduce2 f list,
  - fun List.filter f list,
  - fun List.foldl/foldr f init list
- **Scheme**: (your code must run successfully in DrScheme's "Pretty Big" language level)
  - standard math (+, -, *, /, div, mod, ::, @, ^, o, not, andalso, orelse, <, >, <=, >=, =, <>), numeric functions (e.g. abs and Int.max), list functions (e.g. hd, tl, length, rev, foldl, foldr), conversion functions (real, trunc, floor, ceil, ord, chr, str), string functions (implode, explode, concat, size), standard tuple functions (#1, #2, etc.), and any other functions from ML basic data type structures such as Int, Real, String, Bool, and Char (but not List);
  - fun quicksort(f, list), fun m -- n,
  - fun mapx(f, list), fun map f list,
  - fun filterx(f, list), fun filter2 f list,
  - fun reduce(f, list), fun reduce2 f list,
  - fun List.filter f list,
  - fun List.foldl/foldr f init list
- **JavaScript**: standard operators, print, typeof, parseInt, parseFloat, strings (charAt, charCodeAt, indexOf, join, length, match, replace, slice, split, toLowerCase, toUpperCase), Math (Math.abs, ceil, sin, round, etc.), arrays (concat, filter, indexOf, join, map, pop, push, reduce, reverse, shift, slice, sort, splice, unshift), functions (apply, call, bind), instanceof; regular expressions, prototypes, variadic functions, anonymous functions (lambdas)

You also may call any function that is a problem on this exam, whether or not you correctly solve that problem.

The following are explicitly forbidden unless the problem specifically authorizes you to use them:

- **ML**: mutation; arrays; vectors; while loops
- **Scheme**: eval, mutation (set!, set-car!, mcons, set-mcar!, etc.)
- **JavaScript**: eval; with; the Underscore library or other JavaScript libraries; importing Java classes via Rhino

You don't need to write any use, open, include, load, or other "import" statements in your exam code. Do not abbreviate any code. You can write helper functions if they are defined locally and not at the top level.

Unless this page or the problem mentions otherwise, your code you write will be graded purely on external correctness (proper behavior and output) and not on internal correctness (style). Some functions do have specific style constraints.