

Acknowledgment: This concise list was contributed by Charilaos Skiadas.

Notes on material:

- Clear your mind of any preconceived notions of programming ideas.
- Download the reading notes!
- For strings, you have to use double-quote `\verb|"|"`, and not single-quote `\verb|'|`.
- Unary minus is denoted by a tilde `\verb|~|`.
- Test for equality is a single equals `\verb|=|` instead of double `\verb|==|`.
- Every `\verb|if ... then|if...then` must also have an `\verb|else|else`.
- You need a `\verb|val|val` or `\verb|fun|fun` for anything defined at the "top" level (or anywhere else).
- Always restart the REPL before use-ing the same file twice.
- There are no "assignments", there are "variable bindings". Understand the difference and how shadowing works.
- Types are indicated via a colon *after* the variable -- they are often optional (SML can infer them).
- Forget "for/while loops"; you can do the same with recursion instead.
- Functions are values! (Make sure to understand how it is "evaluated".)
- The type of a tuple `\verb|(23, 34)|(23,34)` is denoted as `\verb|int * int|int*int`.
- Function types are denoted by `\verb|->|->`.
- Need to define a "local" variable in a function? Use `\verb|let ... in ... end|let...in...end`.
- "and" is done via `\verb|andalso|andalso`, "or" via `\verb|orelse|orelse`.
- "not equal" is `\verb|<>|<>`.
- You cannot directly do arithmetic on an `\verb|int|int` and a `\verb|real|real`.

Notes on the assignment:

- Read assignment directions *very* carefully.
- Examine the required types of the functions, found on the "summary" section of the notes, page 2. Understand why those are the types. Make sure your implementations have those types.
- Don't use pattern matching on this assignment.
- Don't use features or library functions not described in the lectures/notes (except for places where the assignment specifies). In particular, you cannot use any of the `\verb|List|List` structure's functions.
- Write tests for your functions!

- If your problem returns a boolean, oftentimes you can get by with just `\verb|andalso|andalso` and `\verb|orelse|orelse`, without needing if-then-else. At the very least, avoid `\verb|if e then true else false|ifethentrueelsefalse`.
- Avoid recomputing a value when possible. Using `\verb|#1|#1`, `\verb|#2|#2`, `\verb|hd|hd`, and `\verb|tl|tl` doesn't really cost anything, but other computations should probably be stored if they are repeated.
- Most functions (with the exception of challenge problems) have bodies in the range of 3-8 lines.
- Make sure your functions are spelled correctly!
- You need to deal with empty lists as a possibility, do not assume they are non-empty.
- In general, do not make assumptions that are not explicitly stated in the assignment (if in doubt, ask in the forums!).
- Take care to think of what "older" and "oldest" means for dates.
- Don't forget to submit in two places! Normal assignment, and peer review.