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 ", and not single-quote ’.
- Unary minus is denoted by a tilde ~.
- Test for equality is a single equals = instead of double ==.
- Every if...then must also have an else.
- You need a val or 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 (23,34) is denoted as int*int.
- Function types are denoted by ->.
- Need to define a "local" variable in a function? Use let...in...end.
- "and" is done via andalso, "or" via orelse.
- "not equal" is <>.
- You cannot directly do arithmetic on an int and a 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 List structure's functions.
- Write tests for your functions!
- If your problem returns a boolean, oftentimes you can get by with just andalso and orelse, without needing if-then-else. At the very least, avoid ifethentrueelsefalse.
- Avoid recomputing a value when possible. Using #1, #2, hd, and 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.