Chapter 1 & 2 outline:

- Introduction, sets and elements (last week Monday)
- Set operations; visual verification of set propositions (last week Wednesday)
- Introduction to SML; cardinality and Cartesian products (last week Friday)
- Making types in SML (this week Wednesday)
- Functions in SML (last week Friday)
- Keep working on functions; begin lists (**Today**)
- Functions on lists (Wednesday)
- Powersets; a language processor (Friday)
- (Begin chapter 3, Propositions, next week Monday)

Today:

- Review of functions principles
- Function examples
- Principles of lists
- Type analysis of lists

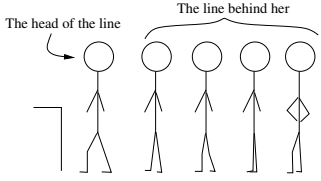
1. Lists must have at least one item.

2. All elements in a list must have the same type.

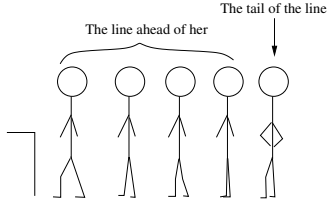
3. Lists can have tuples in them

4. Tuples can have lists in them.

5. This is a good way to think of lists:



6. This is a good way to think of lists:



```
[tl([5, 12, 6])@[8, 9]]
```

```
hd([12, 5, 6])::[2, 7]
```

```
[[(2.3, 5), (8.1, 6)],[]]
```

```
([1, 12, 81], ["a", "bc"])
```

For next time: (This was originally the assignment for today)

Pg 48: 1.11.(4, 8, 10)
Pg 50-51: 1.12.(3, 5, 8)

See assignment notes on Canvas.
Starting with this assignment, HW problems that ask you to write an SML function should be submitted using the "Programming assignment turn-in page." You do **not** *need to include your SML code with your on-paper problems that you turn in.*

No reading or quiz, but it wouldn't hurt to re-read Sections 2.(1-3) before class on Wednesday.