

Chapter 1 & 2 outline:

- ▶ Introduction, sets and elements (last week Wednesday)
- ▶ Set operations; visual verification of set propositions (last week Friday)
- ▶ Introduction to SML; cardinality and Cartesian products (this past Monday)
- ▶ Making types and functions in SML (this past Wednesday)
- ▶ More about functions in SML; introduction to lists [Chapter 2] (today)
- ▶ Functions on lists; powersets (next week Wednesday)
- ▶ Application: A language processor (next week Friday)

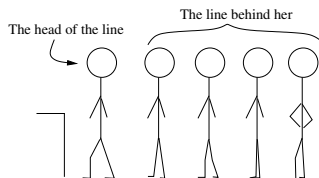
Today:

- ▶ Unfinished business from last time
- ▶ Recursive functions
- ▶ Lists: Definition, operations, types
- ▶ (Time permitting) Functions on lists

1. Lists must have at least one item.

3. Lists can have tuples in them

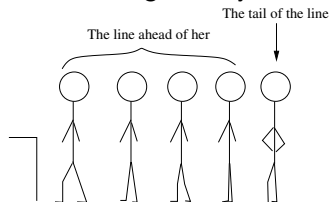
5. This is a good way to think of lists:



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

4. Tuples can have lists in them.

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



[t1([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:

Pg 48: 1.11.(4, 8, 10)

Pg 50-51: 1.12.(3, 5, 8)

Pg 70: 2.1.(2-4, 9, 10)

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

Reread 2.2 (as necessary)

Skim 2.3

Read 2.4