Exercise 2F-2. The error is in the line "Pick any arbitrary $x \in Y \cap Y'$." To be able to pick an element of a set, it must be shown that the set is not empty. In the case where $|X| = 1$, then $Y = Y' = \varnothing$, so it is impossible to pick an $x \in Y \cap Y'$, so the proof of the inductive step is not valid.

Exercise 2F-3. We proceed by structural induction on the derivation tree $D :: \langle \text{while } b \text{ do } x := x+2, \sigma\rangle \Downarrow \sigma$. We have two cases for induction defined by the two inference rules for "while". In the first case (a base case), the last rule used in D was:

$$\frac{\langle b, \sigma\rangle \Downarrow false}{\langle \text{while } b \text{ do } x := x + 2, \sigma\rangle \Downarrow \sigma}$$

Thus $\sigma' = \sigma$, so $\sigma'(x) = \sigma(x)$ so $\sigma'(x)$ is even. In the second case (an inductive case), the last rule used in D was:

$$\frac{\langle b, \sigma\rangle \Downarrow true \quad \text{D'} :: \langle x := x + 2;\ \text{while } b \text{ do } x := x + 2, \sigma\rangle \Downarrow \sigma'}{\langle \text{while } b \text{ do } x := x + 2, \sigma\rangle \Downarrow \sigma'}$$

Now we can simplify to get D' :: $\langle \text{while } b \text{ do } x := x + 2, \sigma[x := \sigma[x] + 2]\rangle \Downarrow \sigma'$, and since D' is smaller than D, we can apply the inductive hypothesis at D' to see that $\sigma'(x)$ is even.

Exercise 1F-4.

$$\frac{}{\langle \text{throw } n, \sigma\rangle \Downarrow \sigma \text{ exc } e}$$

$$\frac{\langle c_1, \sigma\rangle \Downarrow \sigma' \quad \langle c_2, \sigma'\rangle \Downarrow t}{\langle \text{ try } c_1 \text{ catch } x\ c_2, \sigma\rangle \Downarrow t}$$

$$\frac{\langle c_1, \sigma\rangle \Downarrow \sigma' \text{ exc } e \quad \langle c_2, \sigma'[x := e]\rangle \Downarrow t}{\langle \text{ try } c_1 \text{ catch } x\ c_2, \sigma\rangle \Downarrow t}$$

$$\frac{\langle c_1, \sigma\rangle \Downarrow \sigma' \quad \langle c_2, \sigma'\rangle \Downarrow t}{\langle \text{ after } c_1 \text{ finally } c_2, \sigma\rangle \Downarrow t}$$

$$\frac{\langle c_1, \sigma\rangle \Downarrow \sigma' \text{ exc } e_1 \quad \langle c_2, \sigma'\rangle \Downarrow \sigma''}{\langle \text{ after } c_1 \text{ finally } c_2, \sigma\rangle \Downarrow \sigma'' \text{ exc } e_1}$$

$$\frac{\langle c_1, \sigma\rangle \Downarrow \sigma' \text{ exc } e_1 \quad \langle c_2, \sigma'\rangle \Downarrow \sigma'' \text{ exc } e_2}{\langle \text{ after } c_1 \text{ finally } c_2, \sigma\rangle \Downarrow \sigma'' \text{ exc } e_2}$$

Question assigned to the following page: 5

Exercise 1F-5.
Describing "IMP with exceptions" using small-step contextual semantics would be less natural than using big-step semantics because exceptions break the usual flow of computation. In small-step semantics, the next step to take is determined by pattern matching on the current expression, without consideration of whether there has been an exception or not. For a program c0; c1, if c0 throws an exception, then c1 will never be executed, but small-step semantics will require that it is fully reduced before it's result can be discarded. This will especially be a problem if c1 contains an infinite loop because it means small-step semantics will not be able to correctly reduce it.