1 2F-1 Bookkeeping

- **0 pts** Correct

Exercise 2F-2. *Proof:* Let F be the set of all flowers and let $\mathsf{smells}(f)$ be the smell of the flower $f \in F$. (The range of smells is not so important, but we'll assume that it admits equality.) We'll also assume that F is countable. Let the property $P(n)$ mean that all subsets of F of size at most n contain flowers that smell the same.

$$P(n) \stackrel{\text{def}}{=} \forall X \in \mathcal{P}(F).\ |X| \leq n \implies (\forall f, f' \in X.\ \mathsf{smells}(f) = \mathsf{smells}(f'))$$

(the notation $|X|$ denotes the number of elements of X)

One way to formulate the statement to prove is $\forall n \geq 1. P(n)$. We'll prove this by induction on n, as follows:

Base Case: $n = 1$. Obviously all singleton sets of flowers contain flowers that smell the same (by the definition of $P(n)$).

Induction Step: Let n be arbitrary and assume that all subsets of F of size at most n contain flowers that smell the same. We will prove that the same thing holds for all subsets of size at most $n + 1$. Pick an arbitrary set X such that $|X| = n + 1$. Pick two distinct flowers $f, f' \in X$ and let's show that $\mathsf{smells}(f) = \mathsf{smells}(f')$. Let $Y = X - \{f\}$ and $Y' = X - \{f'\}$. Obviously Y and Y' are sets of size at most n so the induction hypothesis holds for both of them. Pick any arbitrary $x \in Y \cap Y'$. Obviously, $x \neq f$ and $x \neq f'$. We have that $\mathsf{smells}(f') = \mathsf{smells}(x)$ (from the induction hypothesis on Y) and $\mathsf{smells}(f) = \mathsf{smells}(x)$ (from the induction hypothesis on Y'). Hence $\mathsf{smells}(f) = \mathsf{smells}(f')$, which proves the inductive step, and the theorem.

The error stems from the assumption that the highlighted portion will generate any x at all. At $n = 2$, $Y \cap Y'$ will result in an empty set since they will each only contain either f or f', causing this induction to fail.

Exercise 2F-3.

We will do an induction on b. As for our $\preccurlyeq$, our base case is a single "false" or a constant "true", and our inductive case is adding "true" before the other cases (to simulate looping - it will either terminate with a "false" in the end, or go on looping indefinitely). This is well-founded since while the number of "false"s can increase indefinitely, it can't decrease any lower than 0, and thus there are no infinite descending chains.

Base case: $b ::= false$. The state terminates without even going through the loop once, and thus x is unchanged. Hence, it will remain even.

Base case: $b ::= constant\ true$. The state never terminates. Hence the given if statement will be false and since false $\Rightarrow$ anything is always true, the entire statement still holds true.

Inductive case: $b ::= true\ then\ b'$, *where* b' *holds true.* since we assume the statement holds true when $b = b'$, we basically only need to prove that $x := x + 2$ is even when x is even. Let $y = x/2$, since x is even, y is an integer. Then $(x + 2)/2 = (2y + 2)/2 = y + 1$, and $y + 1$ is also an integer. Hence $x + 2$ is even, and the statement is proven by induction.

2 2F-2 Mathematical Induction

- **0 pts** Correct

Exercise 2F-2. *Proof:* Let F be the set of all flowers and let $\mathsf{smells}(f)$ be the smell of the flower $f \in F$. (The range of smells is not so important, but we'll assume that it admits equality.) We'll also assume that F is countable. Let the property $P(n)$ mean that all subsets of F of size at most n contain flowers that smell the same.

$$P(n) \stackrel{\text{def}}{=} \forall X \in \mathcal{P}(F).\ |X| \leq n \implies (\forall f, f' \in X.\ \mathsf{smells}(f) = \mathsf{smells}(f'))$$

(the notation $|X|$ denotes the number of elements of X)

One way to formulate the statement to prove is $\forall n \geq 1. P(n)$. We'll prove this by induction on n, as follows:

Base Case: $n = 1$. Obviously all singleton sets of flowers contain flowers that smell the same (by the definition of $P(n)$).

Induction Step: Let n be arbitrary and assume that all subsets of F of size at most n contain flowers that smell the same. We will prove that the same thing holds for all subsets of size at most $n + 1$. Pick an arbitrary set X such that $|X| = n + 1$. Pick two distinct flowers $f, f' \in X$ and let's show that $\mathsf{smells}(f) = \mathsf{smells}(f')$. Let $Y = X - \{f\}$ and $Y' = X - \{f'\}$. Obviously Y and Y' are sets of size at most n so the induction hypothesis holds for both of them. Pick any arbitrary $x \in Y \cap Y'$. Obviously, $x \neq f$ and $x \neq f'$. We have that $\mathsf{smells}(f') = \mathsf{smells}(x)$ (from the induction hypothesis on Y) and $\mathsf{smells}(f) = \mathsf{smells}(x)$ (from the induction hypothesis on Y'). Hence $\mathsf{smells}(f) = \mathsf{smells}(f')$, which proves the inductive step, and the theorem.

The error stems from the assumption that the highlighted portion will generate any x at all. At $n = 2$, $Y \cap Y'$ will result in an empty set since they will each only contain either f or f', causing this induction to fail.

Exercise 2F-3.

We will do an induction on b. As for our $\preccurlyeq$, our base case is a single "false" or a constant "true", and our inductive case is adding "true" before the other cases (to simulate looping - it will either terminate with a "false" in the end, or go on looping indefinitely). This is well-founded since while the number of "false"s can increase indefinitely, it can't decrease any lower than 0, and thus there are no infinite descending chains.

Base case: $b ::= false$. The state terminates without even going through the loop once, and thus x is unchanged. Hence, it will remain even.

Base case: $b ::= constant\ true$. The state never terminates. Hence the given if statement will be false and since false $\Rightarrow$ anything is always true, the entire statement still holds true.

Inductive case: $b ::= true\ then\ b'$, *where* b' *holds true.* since we assume the statement holds true when $b = b'$, we basically only need to prove that $x := x + 2$ is even when x is even. Let $y = x/2$, since x is even, y is an integer. Then $(x + 2)/2 = (2y + 2)/2 = y + 1$, and $y + 1$ is also an integer. Hence $x + 2$ is even, and the statement is proven by induction.

3 2F-3 While Induction

- **0 pts** Correct

Exercise 2F-4.

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

$$\frac{\langle c_1, \sigma\rangle \Downarrow \sigma'}{\langle \mathit{try}\ c_1\ \mathit{catch}\ x\ c_2, \sigma\rangle \Downarrow \sigma'}$$

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

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

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

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

Exercise 2F-5. I am against the claim. The advantage that small-step contextual semantics has is that it is hard to talk about commands whose evaluation does not terminate or are ill-formed or errornous in natural-style operational semantics. However, with the addition of exceptions, the need to care about those commands is eliminated, since the exceptions handling will take care of them. Hence it will be simpler to stick with the natural-style operational semantics.

4 2F-4 Language Features, Large Step

- **0 pts** Correct

Exercise 2F-4.

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

$$\frac{\langle c_1, \sigma\rangle \Downarrow \sigma'}{\langle \mathit{try}\ c_1\ \mathit{catch}\ x\ c_2, \sigma\rangle \Downarrow \sigma'}$$

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

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

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

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

Exercise 2F-5. I am against the claim. The advantage that small-step contextual semantics has is that it is hard to talk about commands whose evaluation does not terminate or are ill-formed or errornous in natural-style operational semantics. However, with the addition of exceptions, the need to care about those commands is eliminated, since the exceptions handling will take care of them. Hence it will be simpler to stick with the natural-style operational semantics.

5 2F-5 Language Features, Analysis

- **0 pts** Correct