(e)

```
┌─ Register ──────────────────────────────────
│ del?:      PERSON
│ langs?:    PLANGUAGE
│ ΔCONFERENCE
├─────────────────────────────────────────────
│ del? ∉ delgates
│ delgates' = delegates ∪ {del?}
│ speaks' =
│ speaks ∪ {lan: LANGUAGE | lan ∈ langs? • del? ↦ lan}
│ official' = official
└─────────────────────────────────────────────
```

Chapter 10

1.

(a) *bookedTo* is a function since it maps rooms to person and for any given room at most one person can book it. A person can book any number of rooms.

(b) The function is partial since not all rooms have been booked.

2.

- ➤ **Line 1:** gives the schema a name.
- ➤ **Line 2:** incorporates the schema Δ*Hotel*; permits reference to state variables before and after this operation.
- ➤ **Line 3:** *p?* is an input variable – the person making the booking.
- ➤ **Line 4:** *r?* is an input variable – the room to be booked.
- ➤ **Line 5:** the room must not already be booked.
- ➤ **Line 6:** the maplet relating the room to the person is included in the new value of the function *bookedTo*.

3.

```
┌─ CancelBooking₀ ────────────────────────
│ ΔHotel
│ p?:     PERSON
│ r?:     ROOM
├──────────────────────────────────────────
│ {r? ↦ p?} ∈ bookedTo
│ bookedTo' = bookedTo \ {r? ↦ p?}
└──────────────────────────────────────────
```

4.

- ➤ **Line 1:** gives schema a name.
- ➤ **Line 2:** incorporates the schema Δ*Hotel*; permits reference to state variables before and after this operation.
- ➤ **Line 3:** *p?* is an input variable – the person making the booking.
- ➤ **Line 4:** *r?* is an input variable – the room to be booked.
- ➤ **Line 5:** the room must already be booked to this person.
- ➤ **Line 6:** the maplet relating the room to the person is removed from the new value of the function *bookedTo*.

5.
(a) and (b)

```
┌─ Sydney2000 ────────────────────────────────
│ participating:   PCOUNTRY
│ events:          PEVENT
│ represents:      PERSON ↛ COUNTRY
│ competesIn:      PERSON ↔ EVENT
│ won:             EVENT ↛ PERSON
├─────────────────────────────────────────────
│ ran represents ⊆ participating
│ ∀ ev: EVENT | ev ∈ dom won • (won ev)
│ competesIn ev
└─────────────────────────────────────────────
```

- ➤ A person can only represent one country. An event has at most one winner.
- ➤ Persons may only represent participating countries. The person who wins an event must have been competing in it.

(c)

```
┌─ JoinGames ─────────────────────────────
│ ΔSydney2000
│ c?:       COUNTRY
├──────────────────────────────────────────
│ c? ∉ participating
│ participating' = participating ∪ {c?}
│ events' = events
│ represents' = represents
│ competesIn' = competesIn
│ won' = won
└──────────────────────────────────────────
```

(d)

```
┌─Win───────────────────────────
│ ΔSydney2000
│ p?:        PERSON
│ ev?:       EVENT
├───────────────────────────────
│ ev? ∈ events
│ ev? ∉ dom won
│ p? competesIn ev?
│ won' = won ∪ {ev? ↦ p?}
│ participating' = participating
│ events' = events
│ represents' = represents
│ competesIn' = competesIn
└───────────────────────────────
```

(e)

```
┌─CountryGolds──────────────────
│ ΞSydney2000
│ c?:        COUNTRY
│ golds!:    ℕ
├───────────────────────────────
│ c? ∈ participating
│ golds! =
│    #{ev: EVENT | ev ∈ dom won ∧
│ won ; represents(ev) = c? • ev}
└───────────────────────────────
```

Chapter 11

1.

```
┌─SecureComputer────────────────
│ Computer
│ password:   PERSON ⇸ PASSWORD
├───────────────────────────────
│ dom password = users
└───────────────────────────────
```

2.

```
┌─SecureInit────────────────────
│ SecureComputer
│ InitComputer
├───────────────────────────────
│ password' = ∅
└───────────────────────────────
```

3.

```
┌─SecureAddUser₀────────────────
│ AddUser₀
├───────────────────────────────
│ password' = password ∪ {p? ↦ dummy}
└───────────────────────────────
```

4.

```
┌─SecureLogin₀──────────────────
│ ΔSecureComputer
│ Login
│ pwd?:    PASSWORD
├───────────────────────────────
│ pwd? = password p?
└───────────────────────────────
```

5.

```
┌─ChangePassword₀───────────────
│ ΔSecureComputer
│ Login
│ old?, new?:        PASSWORD
├───────────────────────────────
│ p? ∈ loggedIn
│ password p? = old?
│ password' = password ⊕ {p? ↦ new?}
└───────────────────────────────
```

Chapter 12

1.

$$u \frown v \quad = \quad \langle \text{London, Amsterdam, Madrid, Paris, Frankfurt} \rangle$$
$$\text{rev}(u \frown v) \quad = \quad \langle \text{Frankfurt, Paris, Madrid, Amsterdam, London} \rangle$$
$$\text{rev } u \quad = \quad \langle \text{Madrid, Amsterdam, London} \rangle$$
$$\text{rev } v \quad = \quad \langle \text{Frankfurt, Paris} \rangle$$
$$\text{rev } v \frown \text{rev } u \quad = \quad \langle \text{Frankfurt, Paris, Madrid, Amsterdam, London} \rangle$$

2.

$$\text{squash}(2..4 \lhd \text{rev}(u \frown v)) = \langle \text{Paris, Madrid, Amsterdam} \rangle$$

3.

$$\text{squash}(4..2 \lhd \text{rev}(u \frown v)) = \langle \rangle$$

159