Questions 1-3

Note: please keep aisle-starter.rkt file open, so you can refer to it while answering the following questions.

Question 1

1 point possible (graded)

Consider the test:

```
(check-expect (aisle? 10) false)
```

We did not add it to `aisle?` function design because:

- ○ The value this test produces is incorrect.
- ● This test is redundant. ✔
- ○ The value passed to `aisle?` by this test is not of type `SeatNum`.
- ○ We forgot about it.

Explanation

It's sufficient to test only 1 midpoint in the `SeatNum` interval for the `aisle?` function, since for all seats exclusively between 1 and 32, `aisle?` produces `false`.

Submit

Answers are displayed within the problem

Question 2

1 point possible (graded)

Now consider this other test:

```
(check-expect (aisle? 33) false)
```

We did not add it to `aisle?` function design because:

- ○ The value this test produces is incorrect.
- ○ This test is redundant.
- ● The value passed to `aisle?` by this test is not of type `SeatNum`. ✔
- ○ We forgot about it.

Explanation

According to the signature, `aisle?` consumes `SeatNum`, and `SeatNum` is a natural number between 1 and 32.

Submit

Answers are displayed within the problem

Question 3

1 point possible (graded)

Suppose you are asked to design another function called `middle?` that produces `true` if the seat is anywhere in the middle of the row (i.e not on the aisle seats). Since this function resembles `aisle?`, you would like to use some of it:

```
;; SeatNum -> Boolean
;; produce true if the given seat number is on the aisle
(check-expect (aisle?  1) true)
(check-expect (aisle? 16) false)
(check-expect (aisle? 32) true)

;(define (aisle? sn) false)  ;stub

;<use template from SeatNum>

(define (aisle? sn)
  (or (= sn 1)
      (= sn 32)))
```

To design the function `middle?`, which of the following would be the same as in the `aisle?` function design (check all that apply)?

- [x] The signature ✔
- [ ] The purpose
- [x] The value produced by the stub ✔
- [x] The template used ✔
- [x] The number of check-expects ✔
- [ ] The body of the function

Explanation

Here is what the function would look like:

```
;; SeatNum -> Boolean
;; produce true if the given seat number is anywhere on the middle of the row
(check-expect (middle?  1) false)
(check-expect (middle? 16) true)
(check-expect (middle? 32) false)

;(define (middle? sn) false)  ;stub
;<use template from SeatNum>

(define (middle? sn)
  (and (not (= sn 1))
       (not (= sn 32))))
```

Note that we still need to have at least 3 tests for this function (for both endpoints and a midpoint of the interval).

Submit

Answers are displayed within the problem