Questions 1-4

Bookmark this page

Recall the data definition for Natural:

```
;; Natural is one of:
;; - 0
;; - (add1 Natural)
;; interp. a natural number
(define N0 0)           ;0
(define N1 (add1 N0))   ;1
(define N2 (add1 N1))   ;2

#;
(define (fn-for-natural n)
  (cond [(zero? n)(...)]
        [else
         (... n
              (fn-for-natural (sub1 n)))]))
```

Questions 1-2

2/2 points (graded)

The factorial of a natural number n is n*n-1*n-2*...1. So factorial of 3 is 3*2*1*1. Let's design a function called `fact` to compute the factorial.

- factorial of 0 is 1
- factorial of n is n*factorial(n-1)

Here is the signature and purpose for `fact`:

```
;; Natural -> Natural
;; compute n * n-1 * n-2 * ... * 1
```

What is the result of each check-expect?

```
(check-expect (fact 0) ____ )
```

1 ✔ **Answer:** 1

1

```
(check-expect (fact 3) ____ )
```

6 ✔ **Answer:** 6

6

Explanation
0 factorial is 1, so `(fact 0)` should produce 1.
3 factorial is 3*2*1 whihc is 6.

Submit Show Answer

Answers are displayed within the problem

Questions 3-4

2/2 points (graded)

Here is the partially completed function body for `fact`:

```
(define (fact n)
    (cond [(zero? n)(1)...)]
          [else
           (2)... n
                  (fact (sub1 n)))]))
```

What should go in (1)...?

- 0
- 1
- empty

What should go in (2)...?

- (•) *
- ( ) +
- ( ) cons

Explanation
As we saw in the examples, `(fact 0)` produces 1, so the base case is 1.
The operation we need to combine the values of n, n-1, n-2, ..., 1 is *.

Submit

Show Answer

Answers are displayed within the problem

Previous Next