Exercise 2.26.

Suppose we define `x` and `y` to be two lists:

```
(define x (list 1 2 3))

(define y (list 4 5 6))
```

What result is printed by the interpreter in response to evaluating each of the following expressions:

```
(append x y)

(cons x y)

(list x y)
```

Answer.

We saw in sention 2.2.1 that `append` is a procedure which takes two lists as arguments and combines their elements to make a new list. Thus, the result printed by the interpreter while evaluating `(append x y)` would be:

```
(1 2 3 4 5 6)
```

Using the substitution model, the expression `(cons x y)` evolves into

```
(cons (list 1 2 3) (list 4 5 6))
```

when evaluated. Remember that `(list 4 5 6)` is equivalent to

```
(cons 4
      (cons 5
            (cons 6 nil)))
```

This further causes the original expression evolve into:

```
(cons (list 1 2 3)
      (cons 4
            (cons 5
                  (cons 6 nil))))
```

which can be expressed in a more fimilar way:

```
(list (list 1 2 3) 4 5 6)
```

Hence, what printed by the interpreter while evaluating `(cons x y)` is:

```
((1 2 3) 4 5 6)
```

Finally, the procedure `list` simply glues a list of elements to form a sequence without any other behavior. So the result printed by the interpreter in response to evaluating `(list x y)` is:

```
((1 2 3) (4 5 6))
```