Part 1: Exercises for Section 3.1 (pages. 202-204):

2
a) This procedure is not finite, since execution of the while loop continues forever.
b) This procedure is not effective, because the step \( m := 1/n \) cannot be performed when \( n = 0 \), which will eventually be the case.
c) This procedure lacks definiteness, since the value of \( i \) is never set.
d) This procedure lacks definiteness, since the statement does not tell whether \( x \) is to be set equal to \( a \) or to \( b \).

8 This is similar to Exercise 7, modified to keep track of the largest even integer we encounter.

```
1: procedure LARGEST EVEN LOCATION(a_1, a_2, \cdots, a_n: integers)
2:    k \leftarrow 0
3:    largest \leftarrow -\infty
4:    for i \leftarrow 1 to n do
5:       if a_i is even and a_i > largest then
6:          k \leftarrow i
7:       largest \leftarrow a_i
8:    return k \{the desired location (or 0 if there are no evens)\}
```
10 We assume that if the input $x = 0$, then $n > 0$, since otherwise $x^n$ is not defined. In our procedure, we let $m = |n|$ and compute $x^m$ in the obvious way. Then if $n$ is negative, we replace the answer by its reciprocal.

```plaintext
1: procedure POWERFCT(x: Real, n: integer)
2:     $m \leftarrow |n|$
3:     power $\leftarrow$ 1
4:     for $i \leftarrow 1$ to $m$ do
5:         power $\leftarrow$ power $\cdot$ $x$
6:     if $n < 0$ then
7:         power $\leftarrow$ 1/power
8: return power \{ power $= x^n$ \}
```
a) With linear search we start at the beginning of the list, and compare 7 successively with 1, 3, 4, 5, 6, 8, 9, and 11. When we come to the end of the list and still have not found 7, we conclude that it is not in the list.

b) We begin the search on the entire list, with $i = 1$ and $j = n = 8$. We set $m := 4$ and compare 7 to the fourth element of the list. Since $7 > 5$, we next restrict the search to the second half of the list, with $i=5$ and $j=8$. This time we set $m := 6$ and compare 7 to the sixth element of the list. Since $7 > 8$, we next restrict ourselves to the first half of the second half of the list, with $i = 5$ and $j = 6$.

This time we set $m := 5$, and compare 7 to the fifth element. Since $7 > 6$, we now restrict ourselves to the portion of the list between $i = 6$ and $j = 6$. Since at this point $i > j$, we exit the loop. Since the sixth element of the list is not equal to 7, we conclude that 7 is not in the list.

16

We let min be the smallest element found so far. At the end, it is the smallest element, since we update it as necessary as we scan through the list.

```plaintext
1: procedure SMALLEST(a_1, a_2, ⋯, a_n: natural numbers)
2:     min ← a_1
3:     for i ← 2 to n do
4:         if a_i < min then
5:             min ← a_i
6:     return min {the smallest integer among the input}
```

We just combine procedures for finding the largest and smallest elements.

1: **procedure** SMALLEST AND LARGEST\((a_1, a_2, \cdots, a_n; \text{ integers})\)
2: \(\text{min} \leftarrow a_1\)
3: \(\text{max} \leftarrow a_1\)
4: \(\textbf{for } i \leftarrow 2 \textbf{ to } n \textbf{ do}\)
5: \(\text{if } a_i < \text{min} \textbf{ then}\)
6: \(\text{min} \leftarrow a_i\)
7: \(\text{if } a_i > \text{max} \textbf{ then}\)
8: \(\text{max} \leftarrow a_i\)
9: \(\textbf{return } \{ \text{min is the smallest integer among the input, and max is the largest} \}\)