64. Each application of the function $f$ divides its argument by 2. Therefore iterating this function $k$ times (which is what $f^{(k)}$ does) has the effect of dividing by $2^k$. Therefore $f^{(k)}(n) = n/2^k$. Now $f^*_1(n)$ is the smallest $k$ such that $f^{(k)}(n) \leq 1$, that is, $n/2^k \leq 1$. Solving this for $k$ easily yields $k \geq \log n$, where logarithm is taken to the base 2. Thus $f^*_1(n) = \lceil \log n \rceil$ (we need to take the ceiling function because $k$ must be an integer).

SECTION 5.4 Recursive Algorithms

2. First, we use the recursive step to write $6! = 6 \cdot 5!$. We then use the recursive step repeatedly to write $5! = 5 \cdot 4!$, $4! = 4 \cdot 3!$, $3! = 3 \cdot 2!$, $2! = 2 \cdot 1!$, and $1! = 1 \cdot 0!$. Inserting the value of $0! = 1$, and working back through the steps, we see that $1! = 1 \cdot 1 = 1$, $2! = 2 \cdot 1! = 2 \cdot 1 = 2$, $3! = 3 \cdot 2! = 3 \cdot 2 = 6$, $4! = 4 \cdot 3! = 4 \cdot 6 = 24$, $5! = 5 \cdot 4! = 5 \cdot 24 = 120$, and $6! = 6 \cdot 5! = 6 \cdot 120 = 720$.

4. First, because $n = 10$ is even, we use the `else if` clause to see that

$$mpower(2, 10, 7) = mpower(2, 5, 7)^2 \mod 7.$$ 

We next use the `else` clause to see that

$$mpower(2, 5, 7) = (mpower(2, 2, 7)^2 \mod 7 \cdot 2 \mod 7) \mod 7.$$ 

Then we use the `else if` clause again to see that

$$mpower(2, 2, 7) = mpower(2, 1, 7)^2 \mod 7.$$ 

Using the `else` clause again, we have

$$mpower(2, 1, 7) = (mpower(2, 0, 7)^2 \mod 7 \cdot 2 \mod 7) \mod 7.$$ 

Finally, using the `if` clause, we see that $mpower(2, 0, 7) = 1$. Now we work backward: $mpower(2, 1, 7) = (1^2 \mod 7 \cdot 2 \mod 7) \mod 7 = 2$, $mpower(2, 2, 7) = 2^2 \mod 7 = 4$, $mpower(2, 5, 7) = (4^2 \mod 7 \cdot 2 \mod 7) \mod 7 = 4$, and finally $mpower(2, 10, 7) = 4^2 \mod 7 = 2$. We conclude that $2^{10} \mod 7 = 2$.

6. With this input, the algorithm uses the `else` clause to find that $gcd(12, 17) = gcd(17 \mod 12, 12) = gcd(5, 12)$. It uses this clause again to find that $gcd(5, 12) = gcd(12 \mod 5, 5) = gcd(2, 5)$, then to get $gcd(2, 5) = gcd(5 \mod 2, 2) = gcd(1, 2)$, and once more to get $gcd(1, 2) = gcd(2 \mod 1, 1) = gcd(0, 1)$. Finally, to find $gcd(0, 1)$ it uses the first step with $a = 0$ to find that $gcd(0, 1) = 1$. Consequently, the algorithm finds that $gcd(12, 17) = 1$.

8. The sum of the first $n$ positive integers is the sum of the first $n - 1$ positive integers plus $n$. This trivial observation leads to the recursive algorithm shown here.

```plaintext
procedure sum of first(n : positive integer)
if n = 1 then return 1
else return sum of first(n - 1) + n
```

10. The recursive algorithm works by comparing the last element with the maximum of all but the last. We assume that the input is given as a sequence.

```plaintext
procedure max(a_1, a_2, \ldots, a_n : integers)
if n = 1 then return a_1
else
    m := max(a_1, a_2, \ldots, a_{n-1})
    if m > a_n then return m
    else return a_n
```