Chapter 19 Recursion

1. A method that calls itself. One or more base cases (the simplest case) are used to stop recursion. Every recursive call reduces the original problem, bringing it increasingly close to a base case until it becomes that case.

2. \( f(n) = 2 \) if \( n = 1 \)
   \( f(n) = 2 \cdot 2^{(n-1)} \) for \( n > 1 \)

3. \( f(n) = x \) if \( n = 1 \)
   \( f(n) = x \cdot x^{(n-1)} \) for \( n > 1 \)

4. \( f(n) = 1 \) if \( n = 1 \)
   \( f(n) = f(n-1) + n \) for \( n > 1 \)

5. six times. (base case factorial(0))

6. 25 times (Why?)

   number of time fib is invoked in fib(0) = 1

   number of time fib is invoked in fib(1) = 1

   number of time fib is invoked in fib(2) =
   1 + number of time fib is invoked in fib(1) + number of time fib is invoked in fib(2)
   = 1 + 1 + 1 = 3

   number of time fib is invoked in fib(3) =
   1 + number of time fib is invoked in fib(1) + number of time fib is invoked in fib(2)
   = 1 + 1 + 3 = 5

   number of time fib is invoked in fib(4) =
   1 + number of time fib is invoked in fib(2) + number of time fib is invoked in fib(3)
   = 1 + 3 + 5 = 9

   number of time fib is invoked in fib(5) =
   1 + number of time fib is invoked in fib(3) + number of time fib is invoked in fib(4)
   = 1 + 5 + 9 = 15

   number of time fib is invoked in fib(6) =
   1 + number of time fib is invoked in fib(4) + number of time fib is invoked in fib(5)
   = 1 + 9 + 15 = 25
7.  (a) The output is 15 \((5 + 4 + 3 + 2 + 1 = 15)\)
(b) 7654321

8.  (a) The output is 5 4 3 2 1
(b) The output is 1 2 3 4 5

9.  (a) n is double. There is no guarantee that \(n \neq 0\) will be eventually false.
(b) Infinite recursion due to new Test() inside the constructor Test().

10. omitted.
11. omitted.
12. an overloaded method with additional parameters.
13. \(2^5 - 1\)
14. • Any recursive methods can be converted into a non-recursive method. (TRUE)
    • Recursive method usually takes more time and memory to execute than non-recursive methods. (TRUE)
    • Recursive methods are always simpler than non-recursive methods. (FALSE)
    • There is always a condition statement in a recursive method to check whether a base case is reached. (TRUE)
15. When a method is invoked, its contents are placed into a stack. If a method is recursively invoked, it is possible that the stack space is exhausted. This causes stack overflow.