

EXAMPLE 2.8 Towers of Hanoi

Three vertical poles are mounted on a platform. A number of disks are provided, all of different sizes, and each with a central hole allowing it to be threaded on to any of the poles. Initially, all of the disks are on pole 1, forming a tower with the largest disk at the bottom and the smallest disk at the top. Only a single disk may be moved at a time, from the top of any tower to the top of another tower, but no larger disk may be moved on top of a smaller disk. The problem is to move the tower of disks from pole 1 to pole 2.

According to legend, this task was originally set for the monks at the monastery of Hanoi. Sixty-four disks were provided. Once the monks completed their task, the universe would come to an end. How long should this take?

Rather than the particular problem of moving the tower of 64 disks from pole 1 to pole 2, it will prove helpful to address the more general problem of moving a tower of n disks from pole *source* to pole *dest*.

To render an integer i to the base r (where r is an integer between 2 and 10):

1. If $i < 0$:
 - 1.1. Render '-'.
 - 1.2. Render $(-i)$ to the base r .
2. If $0 \leq i < r$:
 - 2.1. Let d be the digit corresponding to i .
 - 2.2. Render d .
3. If $i \geq r$:
 - 3.1. Let d be the digit corresponding to $(i \text{ modulo } r)$.
 - 3.2. Render (i/r) to the base r .
 - 3.3. Render d .

Algorithm 2.16 Recursive integer rendering algorithm.

```

static String renderBasedInt (int i, int r) {
// Render i to the base r (where r is an integer between 2 and 10).
    String s;
    if (i < 0) {
        s = '-' + renderBasedInt(-i, r);
    } else if (i < r) {
        char d = (char)('0' + i);
        s = "" + d;
    } else {
        char d = (char)('0' + i%r);
        s = renderBasedInt(i/r, r) + d;
    }
    return s;
}

```

Program 2.17 Java implementation of the recursive integer rendering algorithm.