Data Structures and Algorithms 120

CURTIN UNIVERSITY (CRICOS number: 00301J) Faculty of Engineering and Science Department of Computing Data Structures and Algorithms 120 (Index No. 4547/5396)

Practical 3

Aims

- To practice recursion
- To implement Towers of Hanoi

Before the Practical:

• Read this practical sheet fully before starting.

Activity 1: Towers of Hanoi

Implement the Towers of Hanoi algorithm as a java method. Now write a main method to test this. You will also need to implement the moveDisk(src, dest) method. This can be as simple as an output statement printing "Moving top disk from peg source to peg destination" where source and destination are 1, 2 or 3.

Activity 2:

Modify your Towers of Hanoi algorithm so that it additionally informs the user of the import parameters, the (possible) value of temp, and the line or statement number where it was called for every iteration. To make this readable, you must indent each level of recursion. You MUST use spaces, not tabs.

Hint: You might need to add another import parameter. Use the format at appendix 1 of the worksheet as a guide.

Submission Deliverable:

Your modified Towers of Hanoi (including main and moveDisk) program is <u>due at the beginning of your Week 5 tutorial</u> (after the break).

SUBMIT ELECTRONICALLY VIA BLACKBOARD, under the Assessments section.

If you finish early, use the rest of the practical to start the next worksheet, because that will be due later on.

Page 1 of 3 Last Updated: 16/08/14

Appendix 1: Sample output for n=5

```
towers(5, 1, 3)
       n=5, src=1, dest=3, temp=2
       towers(4, 1, 2)
(3)
           n=4, src=1, dest=2, temp=3
(3)
           towers(3, 1, 3)
               n=3, src=1, dest=3, temp=2
               towers(2, 1, 2)
(3)
                   n=2, src=1, dest=2, temp=3
                   towers(1, 1, 3)
(3)
                       n=1, src=1, dest=3
                       Move top disk from peg 1 to peg 3
(1)
                   Move top disk from peg 1 to peg 2
(4)
                   towers(1, 3, 2)
(5)
                       n=1, src=3, dest=2
                       Move top disk from peg 3 to peg 2
(1)
(4)
               Move top disk from peg 1 to peg 3
(5)
               towers(2, 2, 3)
                   n=2, src=2, dest=3, temp=1
(3)
                   towers(1, 2, 1)
                       n=1, src=2, dest=1
                       Move top disk from peg 2 to peg 1
(1)
                   Move top disk from peg 2 to peg 3
(4)
(5)
                   towers(1, 1, 3)
                       n=1, src=1, dest=3
                       Move top disk from peg 1 to peg 3
(1)
(4)
           Move top disk from peg 1 to peg 2
           towers(3, 3, 2)
(5)
               n=3, src=3, dest=2, temp=1
               towers(2, 3, 1)
(3)
                   n=2, src=3, dest=1, temp=2
                   towers(1, 3, 2)
(3)
                       n=1, src=3, dest=2
                       Move top disk from peg 3 to peg 2
(1)
                   Move top disk from peg 3 to peg 1
(4)
(5)
                   towers(1, 2, 1)
                       n=1, src=2, dest=1
                       Move top disk from peg 2 to peg 1
(1)
(4)
               Move top disk from peg 3 to peg 2
(5)
               towers(2, 1, 2)
                   n=2, src=1, dest=2, temp=3
(3)
                   towers(1, 1, 3)
                       n=1, src=1, dest=3
                       Move top disk from peg 1 to peg 3
(1)
(4)
                   Move top disk from peg 1 to peg 2
(5)
                   towers(1, 3, 2)
                       n=1, src=3, dest=2
(1)
                       Move top disk from peg 3 to peg 2
      Move top disk from peg 1 to peg 3
(4)
       towers(4, 2, 3)
(5)
           n=4, src=2, dest=3, temp=1
```

Data Structures and Algorithms 120

```
(3)
           towers(3, 2, 1)
               n=3, src=2, dest=1, temp=3
               towers(2, 2, 3)
(3)
                   n=2, src=2, dest=3, temp=1
(3)
                   towers(1, 2, 1)
                       n=1, src=2, dest=1
(1)
                       Move top disk from peg 2 to peg 1
(4)
                   Move top disk from peg 2 to peg 3
                   towers(1, 1, 3)
(5)
                       n=1, src=1, dest=3
(1)
                       Move top disk from peg 1 to peg 3
(4)
               Move top disk from peg 2 to peg 1
(5)
               towers(2, 3, 1)
                   n=2, src=3, dest=1, temp=2
                   towers(1, 3, 2)
(3)
                       n=1, src=3, dest=2
                       Move top disk from peg 3 to peg 2
(1)
(4)
                   Move top disk from peg 3 to peg 1
                   towers(1, 2, 1)
(5)
                       n=1, src=2, dest=1
(1)
                       Move top disk from peg 2 to peg 1
(4)
           Move top disk from peg 2 to peg 3
           towers(3, 1, 3)
(5)
               n=3, src=1, dest=3, temp=2
               towers(2, 1, 2)
(3)
                   n=2, src=1, dest=2, temp=3
(3)
                   towers(1, 1, 3)
                       n=1, src=1, dest=3
(1)
                       Move top disk from peg 1 to peg 3
(4)
                   Move top disk from peg 1 to peg 2
(5)
                   towers(1, 3, 2)
                       n=1, src=3, dest=2
                       Move top disk from peg 3 to peg 2
(1)
(4)
               Move top disk from peg 1 to peg 3
               towers(2, 2, 3)
(5)
                   n=2, src=2, dest=3, temp=1
                   towers(1, 2, 1)
(3)
                       n=1, src=2, dest=1
                       Move top disk from peg 2 to peg 1
(1)
                   Move top disk from peg 2 to peg 3
(4)
(5)
                   towers(1, 1, 3)
                       n=1, src=1, dest=3
(1)
                       Move top disk from peg 1 to peg 3
```

Page 3 of 3 Last Updated: 16/08/14