McMaster University

Principles of Programming Assignments

Assignment 3 COMPCS 2S03

Assignment Submission Guidelines:

Individually submit through dropbox on avenue.

1. Submit programs for all problems as separate files in a zip folder.
2. Submit research and discussion part as a single file.
3. Code should contain proper COMMENTS. First comment must include your authorship. //Author: Full Name ID

The goal of this assignments is to give you an understanding of the two programming techniques Iteration and Recursion.

Problem 1: (Fibonacci) the Fibonacci series 0, 1, 1, 2, 3, 5, 8, 13, 21, …begins with the terms 0 and 1 and has the property that each succeeding term is the sum of the two preceding terms.

Task 1 (Marks 2): Write a program that calculates the nth Fibonacci number using iterative approach.
Task 2 (Marks 2): Compare this program with the one was done in class using recursion (section 5.15 of book). Compare both solutions in terms of speed, size of code and memory. Which is more suitable for this problem? Reserach online or other resources to support your point of view and give references accordingly.

Problem 2: (Towers of Hanoi) Every budding computer scientist must grapple with certain classic problems, and the Towers of Hanoi is one of the most famous of these. Legend has it that in a temple in the Far East, priests are attempting to move a stack of disks from one peg to another. The initial stack had 64 disks threaded onto one peg and arranged from bottom to top by decreasing size. The priests are attempting to move the stack from this peg to a second peg under the constraints that exactly one disk is moved at a time, and at no time may a larger disk be placed above a smaller disk. A third peg is available for temporarily holding the disks. Supposedly the world will end when the priests complete their task, so there's little incentive for us to facilitate their efforts.
Let's assume that the priests are attempting to move the disks from peg 1 to peg 3. We wish to develop an algorithm that will print the precise sequence of disk-to-disk peg transfers.
If we were to approach this problem with conventional methods, we'd rapidly find ourselves hopelessly knotted up in managing the disks. Instead, if we attack the problem with recursion in mind, it immediately becomes tractable. Moving n disks can be viewed in terms of moving only n – 1 disks (and hence the recursion) as follows:

a) Move n – 1 disks from peg 1 to peg 2, using peg 3 as a temporary holding area.
b) Move the last disk (the largest) from peg 1 to peg 3.
c) Move the n – 1 disks from peg 2 to peg 3, using peg 1 as a temporary holding area.

The process ends when the last task involves moving n = 1 disk, i.e., the base case. This is accomplished by trivially moving the disk without the need for a temporary holding area.

Task 1 Marks 2: Write a program to solve the Towers of Hanoi problem. Use a **recursive** function with four parameters:

a) The number of disks to be moved
b) The peg on which these disks are initially threaded
c) The peg to which this stack of disks is to be moved
d) The peg to be used as a temporary holding area

Your program should print the precise instructions it will take to move the disks from the starting peg to the destination peg. For example, to move a stack of three disks from peg 1 to peg 3, your program should print the following series of moves:

1 → 3 (This means move one disk from peg 1 to peg 3.)

1 → 2

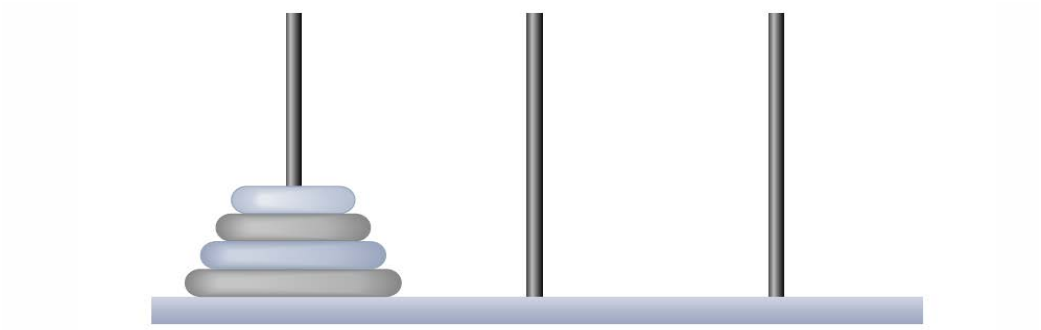
3 → 2

1 → 3

2 → 1

2 → 3

1 → 3



Towers of Hanoi for the case with four disks.

Task 2 Marks 2: Write a program to solve the Towers of Hanoi problem using iteration.

Task 3 Marks 2: Compare both solutions in terms of speed, size of code and memory. Which is more suitable for this problem? Research online or other resources to support your point of view and give references accordingly.