Fin Martinez
CS 4050
Professor Alsaffar
Assignment 1: Tower of Hanoi
9/16/2024

Documentation: Tower of Hanoi

Definition of the Problem

Develop a program to solve the “Tower of Hanoi” problem taking a user provided integer for the number of disks in the problem. The number of rods is constant at 3. User’s can choose to run the program and provide the number of disks, or exit the program.
This problem needs to solve the tower recursively (multi-branched recursion), provide the time taken to solve the problem, and the number of disks used for the problem.

Inputs

For initial menu:
Solve Tower of Hanoi (input number of disks)
Exit program

Outputs

Solver Tower of Hanoi (Time taken to solve problem with number disks and duration)
Exit program upon request

Pseudocode

TowerofHanoi(class):
    Main:
        1. Initialize Scanner, hanoiProcessor, and exitProgram(false)
        2. Loop (switch):
            a. Display program title and user options (Solve Tower of Hanoi, Exit)
            b. Accept operation choice from user (1 or 2)
            c. If selection is 1:
                i. Obtain user input, pass to validation checker
                    1. proceed if valid
                ii. Start system clock
                iii. Start solveTowerOfHanoi with user input parameter
                iv. Stop system clock
                v. Calculate duration
                vi. Display “Time taken to solve the Tower of Hanoi puzzle with <number of disks> disks: <duration> milliseconds”
                vii. Repeat loop
            d. If selection is 2:
                i. Exit program
            e. Display error if input is invalid
                i. Repeat loop

3. Close scanner

getValidNumberOfDisks:

1. Receive user input from Main
2. Set number of disks = 0
3. Set valid input = false
4. Loop:
    a. Prompt user for input
    b. Receive user input
    c. Validate user input
        i. If number of disks greater than 0:
            1. Return true, exit loop
        ii. If number of disks less than 0 or not an integer:
            1. Throw error; repeat loop
    d. If input invalid:
        i. Display error
        ii. Clear invalid input
        iii. Repeat loop
    e. If input valid, return number of disks

HanoiProcessor(class):

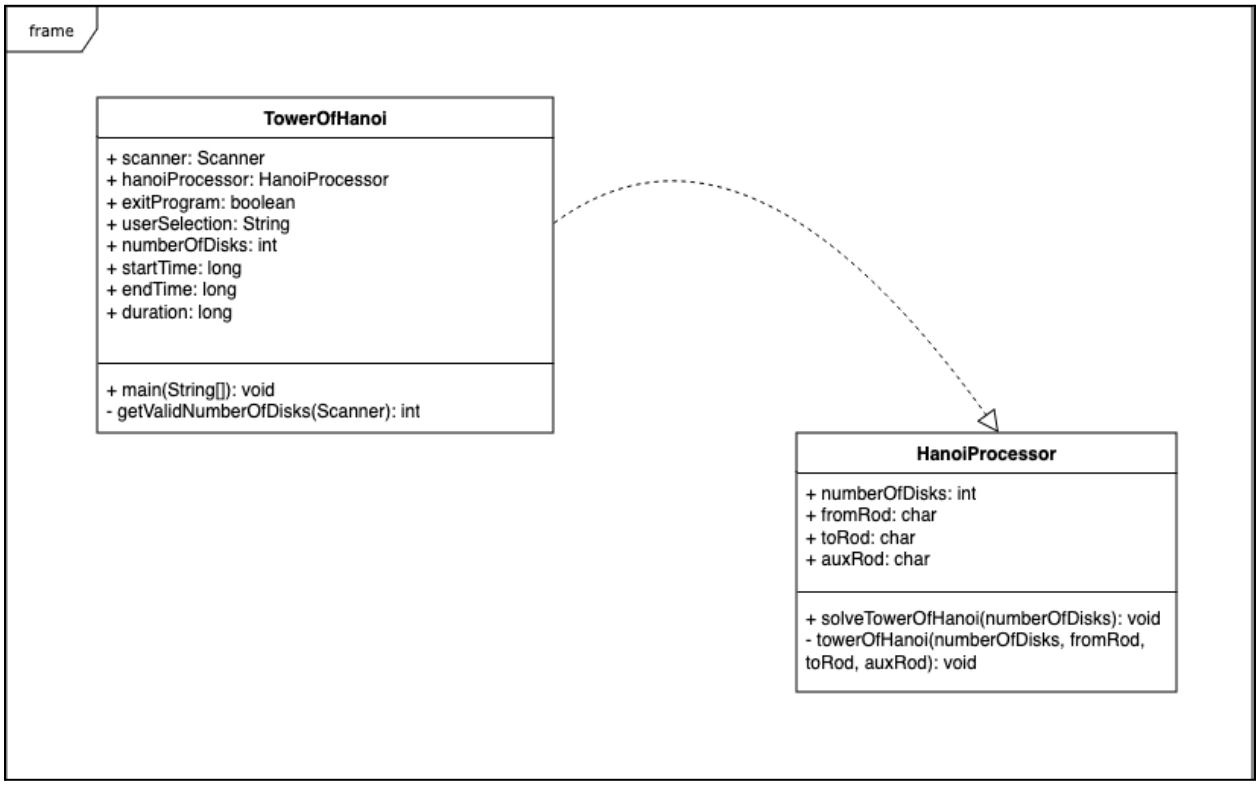
solveTowerOfHanoi:

1. Display “Solving Tower of Hanoi with <number of disks> disks…”
2. Call towerOfHanoi with number of disks (int), and rods as parameters

towerOfHanoi:

1. Set exit parameter (number of disks == 1):
    a. Display “Move disk 1 from rod <source rod> to rod <destination rod>”
    b. Exit recursion when number of disks == 1 and return result
2. Move disks from source rod to auxiliary rod, using destination as buffer
    a. Subtract 1 from number of disks
3. Move largest disk from source to destination
    a. Display (Move disk <number of disks> from <source rod> to <destination rod>
4. Move disks from auxiliary to destination using source rod + disk as buffer
    a. Subtract 1 from number of disks

UML


frame
TowerOfHanoi
+ scanner: Scanner
+ hanoiProcessor: HanoiProcessor
+ exitProgram: boolean
+ userSelection: String
+ numberOfDisks: int
+ startTime: long
+ endTime: long
+ duration: long
+ main(String[]): void
- getValidNumberOfDisks(Scanner): int
HanoiProcessor
+ numberOfDisks: int
+ fromRod: char
+ toRod: char
+ auxRod: char
+ solveTowerOfHanoi(numberOfDisks): void
- towerOfHanoi(numberOfDisks, fromRod, toRod, auxRod): void