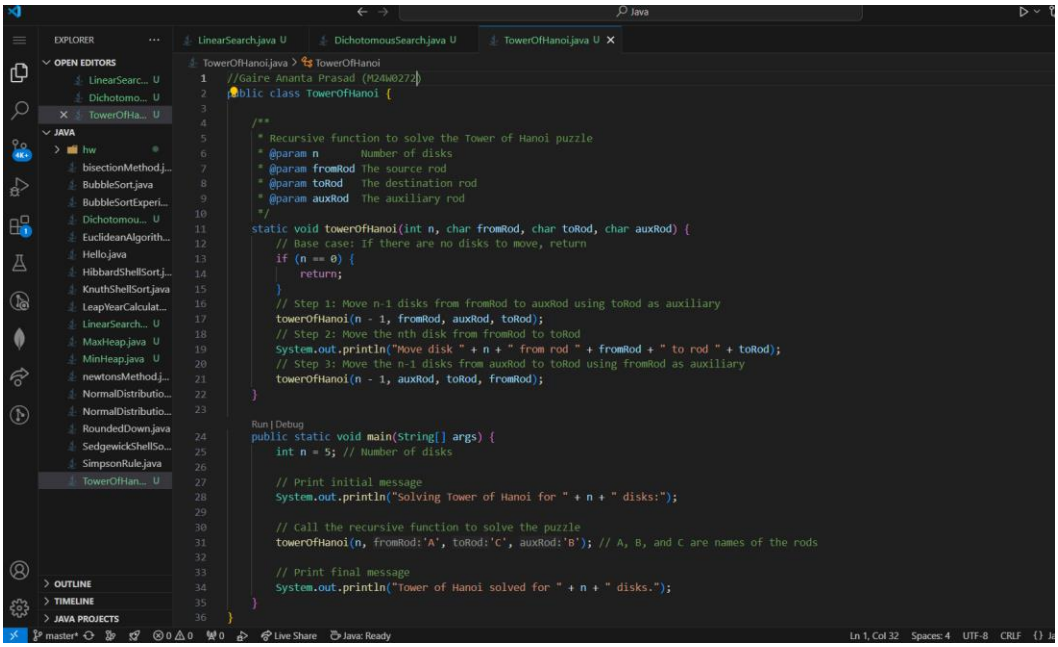
GAIRE ANANTA PRASAD

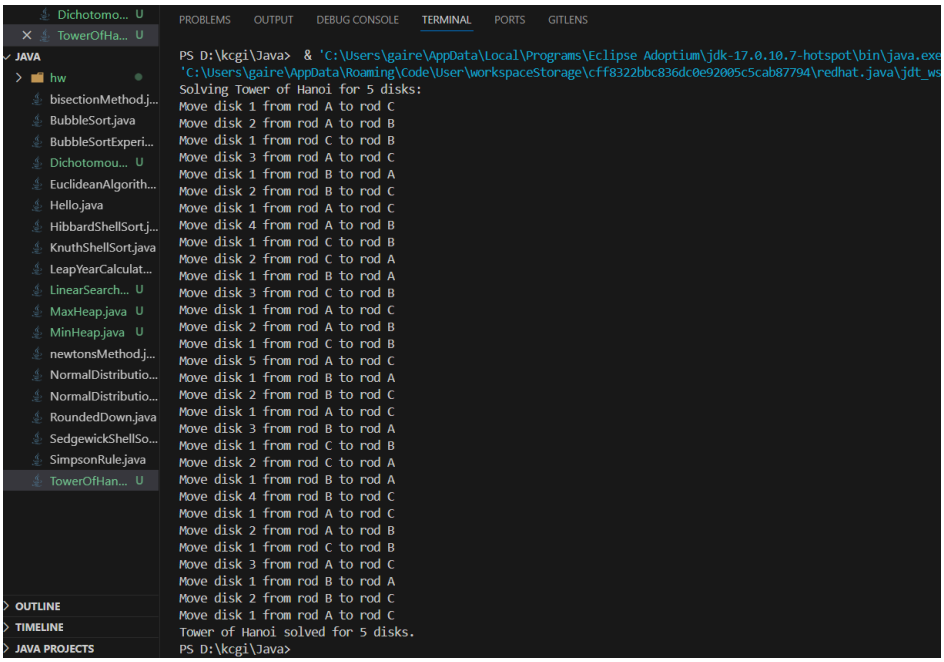
M24W0272

TowerOfHonoi



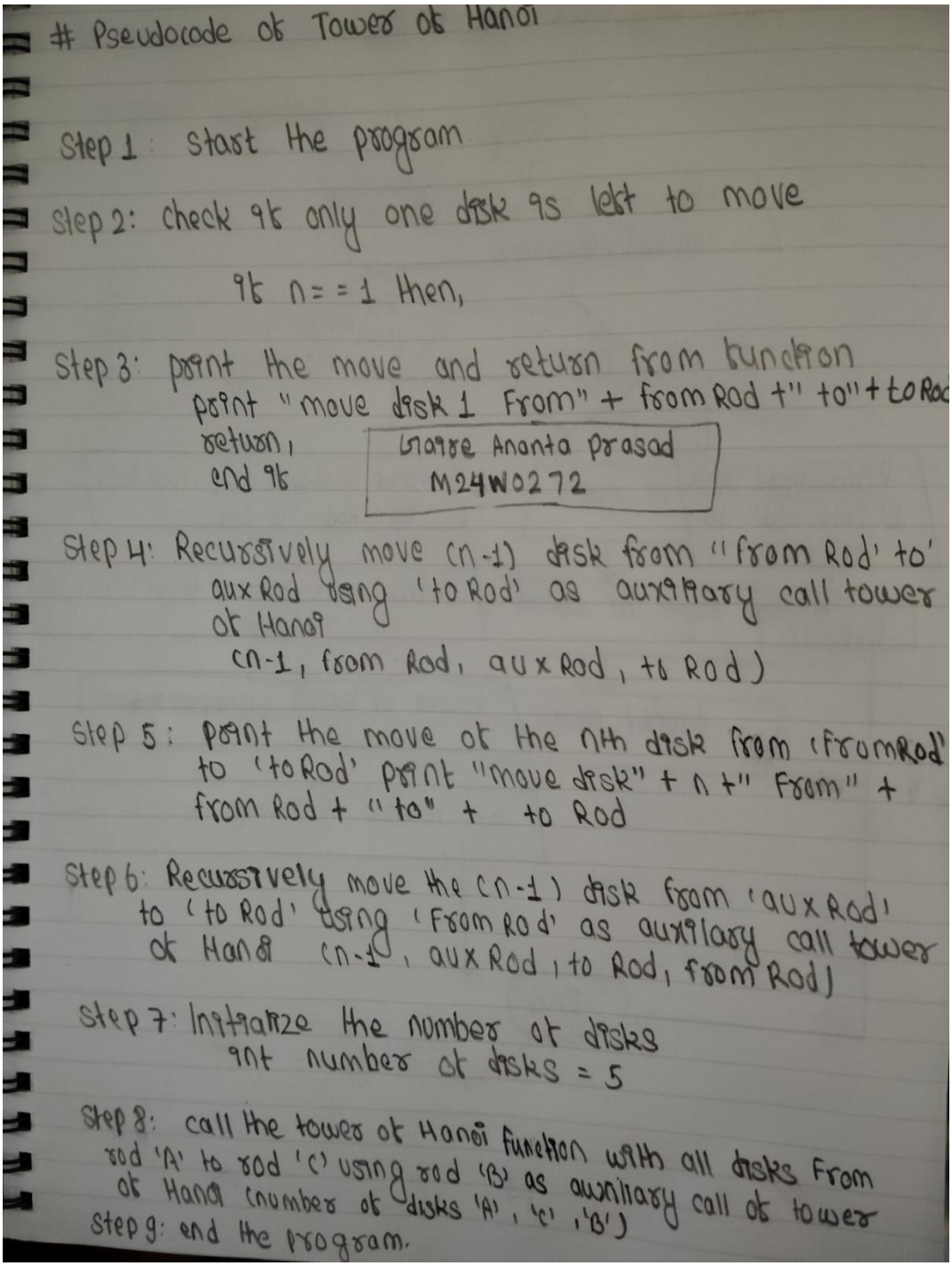
```java
//Gaire Ananta Prasad (M24W0272)
public class TowerOfHanoi {

    /**
     * Recursive function to solve the Tower of Hanoi puzzle
     * @param n       Number of disks
     * @param fromRod The source rod
     * @param toRod   The destination rod
     * @param auxRod  The auxiliary rod
     */
    static void towerOfHanoi(int n, char fromRod, char toRod, char auxRod) {
        // Base case: If there are no disks to move, return
        if (n == 0) {
            return;
        }
        // Step 1: Move n-1 disks from fromRod to auxRod using toRod as auxiliary
        towerOfHanoi(n - 1, fromRod, auxRod, toRod);
        // Step 2: Move the nth disk from fromRod to toRod
        System.out.println("Move disk " + n + " from rod " + fromRod + " to rod " + toRod);
        // Step 3: Move the n-1 disks from auxRod to toRod using fromRod as auxiliary
        towerOfHanoi(n - 1, auxRod, toRod, fromRod);
    }

    Run | Debug
    public static void main(String[] args) {
        int n = 5; // Number of disks

        // Print initial message
        System.out.println("Solving Tower of Hanoi for " + n + " disks:");

        // Call the recursive function to solve the puzzle
        towerOfHanoi(n, fromRod:'A', toRod:'C', auxRod:'B'); // A, B, and C are names of the rods

        // Print final message
        System.out.println("Tower of Hanoi solved for " + n + " disks.");
    }
}
```

Output

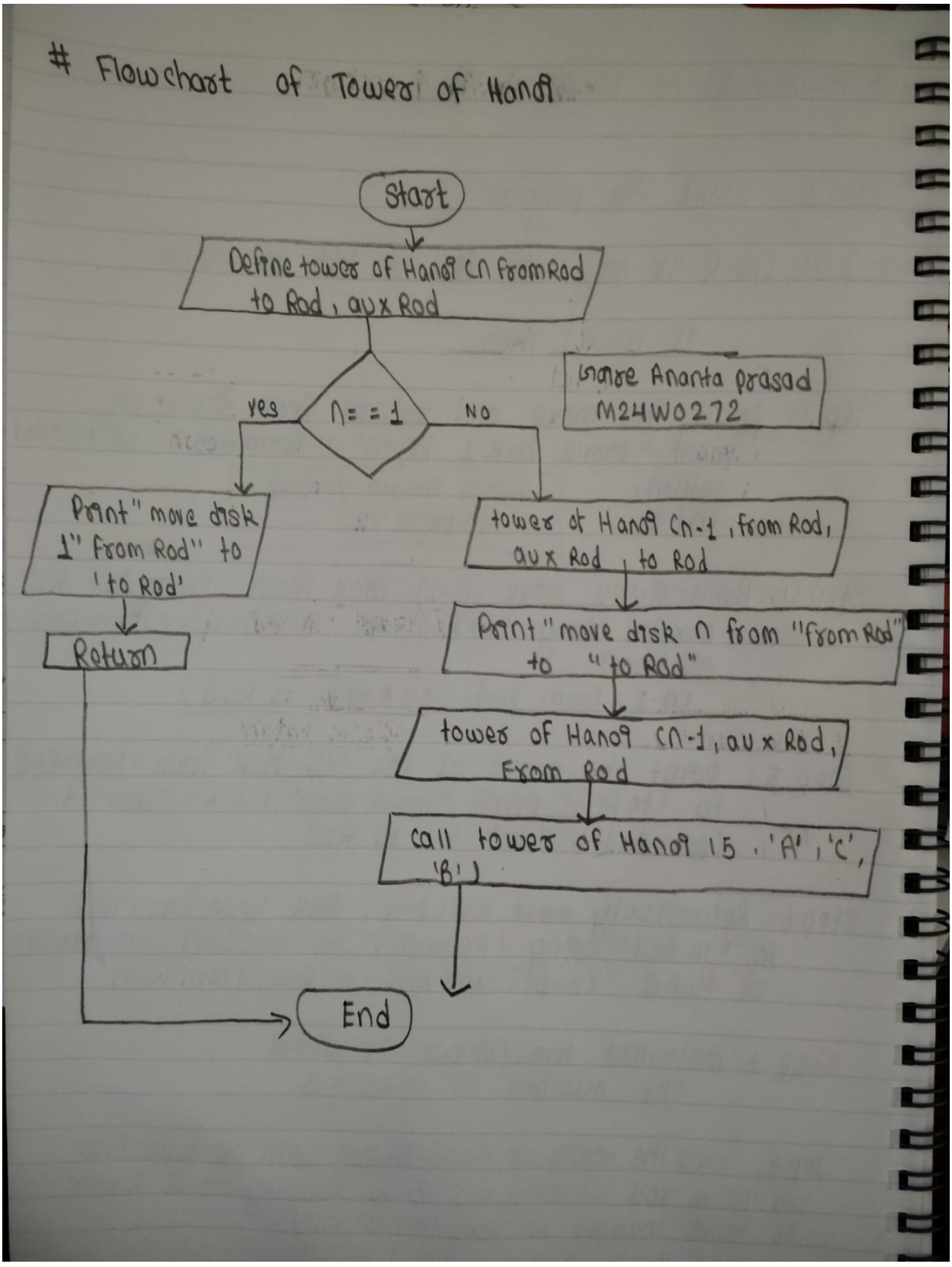


```
PS D:\kcgi\Java> & 'C:\Users\gaire\AppData\Local\Programs\Eclipse Adoptium\jdk-17.0.10.7-hotspot\bin\java.exe
'C:\Users\gaire\AppData\Roaming\Code\User\workspaceStorage\cff8322bbc836dc0e92005c5cab87794\redhat.java\jdt_ws
Solving Tower of Hanoi for 5 disks:
Move disk 1 from rod A to rod C
Move disk 2 from rod A to rod B
Move disk 1 from rod C to rod B
Move disk 3 from rod A to rod C
Move disk 1 from rod B to rod A
Move disk 2 from rod B to rod C
Move disk 1 from rod A to rod C
Move disk 4 from rod A to rod B
Move disk 1 from rod C to rod B
Move disk 2 from rod C to rod A
Move disk 1 from rod B to rod A
Move disk 3 from rod C to rod B
Move disk 1 from rod A to rod C
Move disk 2 from rod A to rod B
Move disk 1 from rod C to rod B
Move disk 5 from rod A to rod C
Move disk 1 from rod B to rod A
Move disk 2 from rod B to rod C
Move disk 1 from rod A to rod C
Move disk 3 from rod B to rod A
Move disk 1 from rod C to rod B
Move disk 2 from rod C to rod A
Move disk 1 from rod B to rod A
Move disk 4 from rod B to rod C
Move disk 1 from rod A to rod C
Move disk 2 from rod A to rod B
Move disk 1 from rod C to rod B
Move disk 3 from rod A to rod C
Move disk 1 from rod B to rod A
Move disk 2 from rod B to rod C
Move disk 1 from rod A to rod C
Tower of Hanoi solved for 5 disks.
PS D:\kcgi\Java>
```

Algorithm


Pseudocode of Tower of Hanoi
Step 1: start the program
Step 2: check if only one disk is left to move
if n == 1 then,
Step 3: print the move and return from function
print "move disk 1 from" + fromRod + "to" + toRod
return,
end if
Gaire Ananta prasad
M24W0272
Step 4: Recursively move (n-1) disk from 'fromRod' to 'auxRod' using 'toRod' as auxiliary call tower of Hanoi
(n-1, fromRod, auxRod, toRod)
Step 5: print the move of the nth disk from 'fromRod' to 'toRod' print "move disk" + n + "from" + fromRod + "to" + toRod
Step 6: Recursively move the (n-1) disk from 'auxRod' to 'toRod' using 'fromRod' as auxiliary call tower of Hanoi (n-1, auxRod, toRod, fromRod)
Step 7: Initialize the number of disks
int number of disks = 5
Step 8: call the tower of Hanoi function with all disks from rod 'A' to rod 'C' using rod 'B' as auxiliary call of tower of Hanoi (number of disks 'A', 'C', 'B')
Step 9: end the program.

Pseudocode



Gaire Ananta Prasad M24W0272