Hands-on Activity 2.1: The Tower of Hanoi Problem

Code

```
d = 4
f = 'A'
to = 'B'
via = 'C'

def hanoiSol(d, f, to, via):
    if d == 1:
        print("Move disk 1 from", f, "to", to)
    else:
        hanoiSol(d-1, f, via, to)
        print("Move disk", d, "from", f, "to", to)
        hanoiSol(d-1, via, to, f)

hanoiSol(d, f, via, to)
```

```
Move disk 1 from A to B
Move disk 2 from A to C
Move disk 1 from B to C
Move disk 3 from A to B
Move disk 1 from C to A
Move disk 2 from C to B
Move disk 1 from A to B
Move disk 4 from A to C
Move disk 1 from B to C
Move disk 2 from B to A
Move disk 1 from C to A
Move disk 3 from B to C
Move disk 1 from A to B
Move disk 2 from A to C
Move disk 1 from B to C
```

Conclusion

The activity focus on creating a program that solves the Tower of Hanoi problem. It highlighted the use of recursion and dynamic programming, which involves using logical and algorithmic thinking in solving the problem. The solution I made particularly involved the recursion method, where the function called itself over and over again until the if condition was followed. In this case, the if statement says that the disk number must be equal to 1, which represents the smallest ring disk must be placed on top of all the

larger disks. If this is not met, the function repeatedly calls itself, which represents the arranging of the large rings in order in the destination pole. This recursive technique helps us solve the problem without individually calling the steps.