Name : Shrinivas Hatyalikar

Div: TY-B (B2)

Roll No: 26

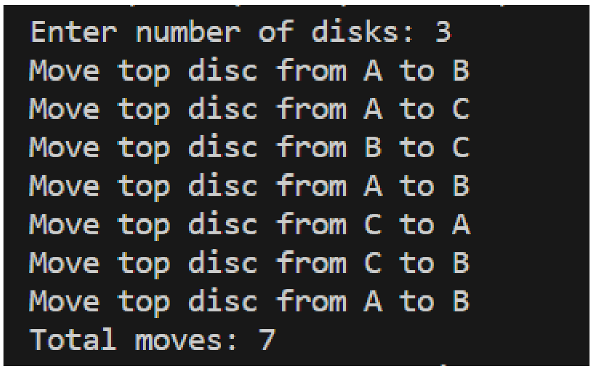
Q) Implementation of Tower Of Hanoi

Part 1:
Input: n (denotes the number of disks)
Output: Assuming that the disks are placed on peg A, print the steps / moves on how to move them to peg C, by following the rules discussed.

Code:

```
#include <stdio.h>

int move_counter = 0;

void move(int n, char source_tower, char destination_tower, char
intermediate_tower) {
    if (n > 0) {
        move(n - 1, source_tower, intermediate_tower, destination_tower);
        printf("Move top disc from %c to %c\n", source_tower,
destination_tower);
        move_counter++;
        move(n - 1, intermediate_tower, destination_tower, source_tower);
    }
}

int main() {
    int n=0;
    printf("Enter number of disks: ");
    scanf("%d",&n);
    move(n, 'A', 'B', 'C');
    printf("Total moves: %d\n", move_counter);
    return 0;
}
```

Output:



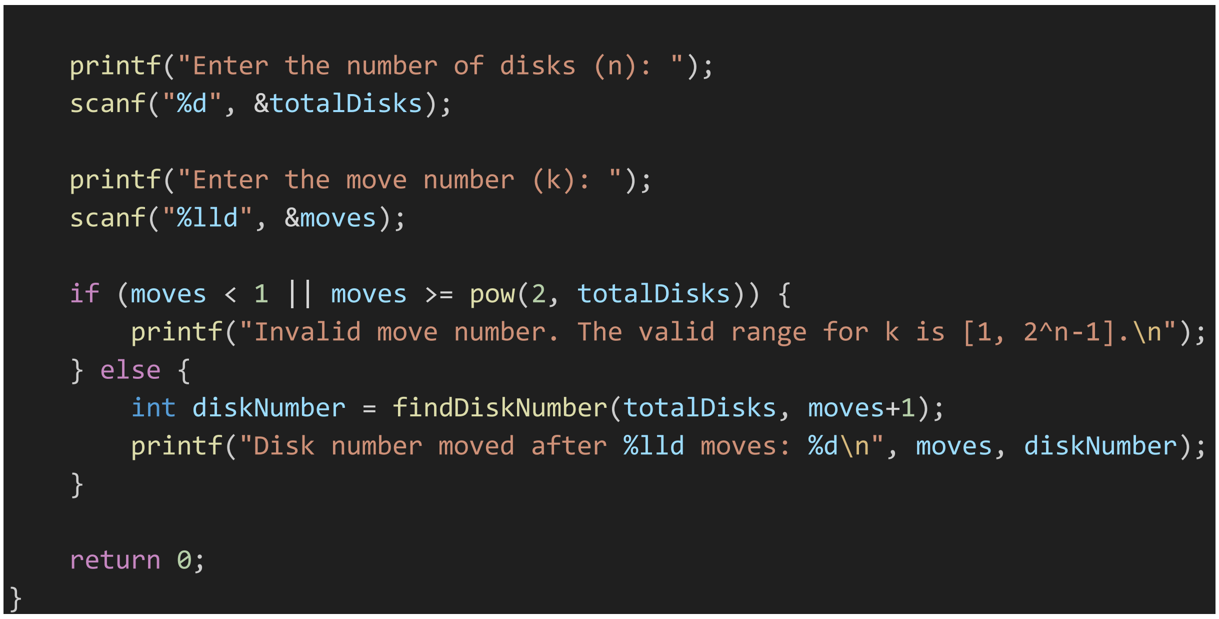
```
Enter number of disks: 3
Move top disc from A to B
Move top disc from A to C
Move top disc from B to C
Move top disc from A to B
Move top disc from C to A
Move top disc from C to B
Move top disc from A to B
Total moves: 7
```

Part 2:
Input: n (denotes the number of disks), k (denotes the move / step number, starts with 1, max value of k is 2^(n)-2

Output: which disk will be moved after the kth move. Assume that disks are numbered from 0 to n-1, 0 being the smallest one.

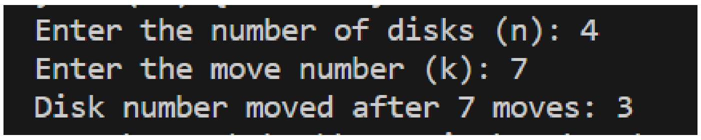
Code:

```c
#include <stdio.h>
#include <math.h>

// Function to find the disk number moved after k moves.
int findDiskNumber(int totalDisks, long long moves) {
    if (totalDisks == 1) {
        return 0; // There is only one disk, so it's disk 0.
    }

    long long mid = pow(2, totalDisks - 1);

    if (moves < mid) {
        return findDiskNumber(totalDisks - 1, moves);
    } else if (moves == mid) {
        return totalDisks - 1; // The largest disk is being moved (disk
totalDisks - 1).
    } else {
        return findDiskNumber(totalDisks - 1, moves - mid);
    }
}

int main() {
    int totalDisks;
    long long moves;
```

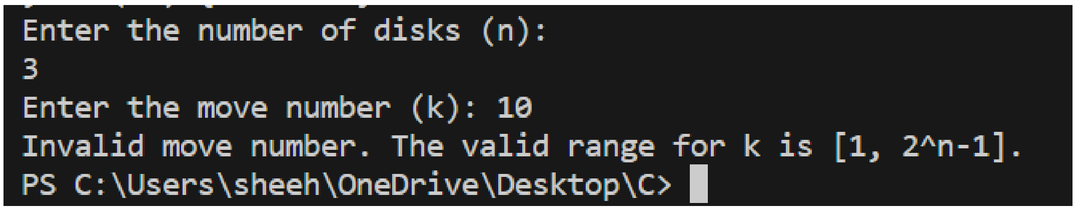


```c
    printf("Enter the number of disks (n): ");
    scanf("%d", &totalDisks);

    printf("Enter the move number (k): ");
    scanf("%lld", &moves);

    if (moves < 1 || moves >= pow(2, totalDisks)) {
        printf("Invalid move number. The valid range for k is [1, 2^n-1].\n");
    } else {
        int diskNumber = findDiskNumber(totalDisks, moves+1);
        printf("Disk number moved after %lld moves: %d\n", moves, diskNumber);
    }

    return 0;
}
```

Output:



```
Enter the number of disks (n): 4
Enter the move number (k): 7
Disk number moved after 7 moves: 3
```



```
Enter the number of disks (n):
3
Enter the move number (k): 10
Invalid move number. The valid range for k is [1, 2^n-1].
PS C:\Users\sheeh\OneDrive\Desktop\C>
```