Question 1.2

```cpp
#include <iostream>
using namespace std;
void Toh(int n, char sourceT, char destiny,
char auxT1, char auxT2){
    if (n == 0){
        return;
    }
    if (n == 1){

    cout <<"T"<< sourceT <<" --> T"<<destiny<<endl;
    return;
    }
    Toh(n - 2, sourceT, auxT1, auxT2,destiny);
    cout <<"T"<< sourceT <<" --> T"<<auxT2<<endl;
    cout <<"T"<< sourceT <<" --> T"<<destiny<<endl;
    cout <<"T"<< auxT2 <<" --> T"<<destiny<<endl;
    Toh(n - 2, auxT1, destiny, sourceT,auxT2);
```

If n = 8, the iteration will work as follows.

Originally called toh(8, '1', '4', '2', '3').

It calls toh(6, '1', '2', '3', '4') multiple times in toh.

It will call Toh(4, '1', '3', '4', '2') again in the next recursive call.

This process continues until Toh(1, '1', '3', '4', '2').

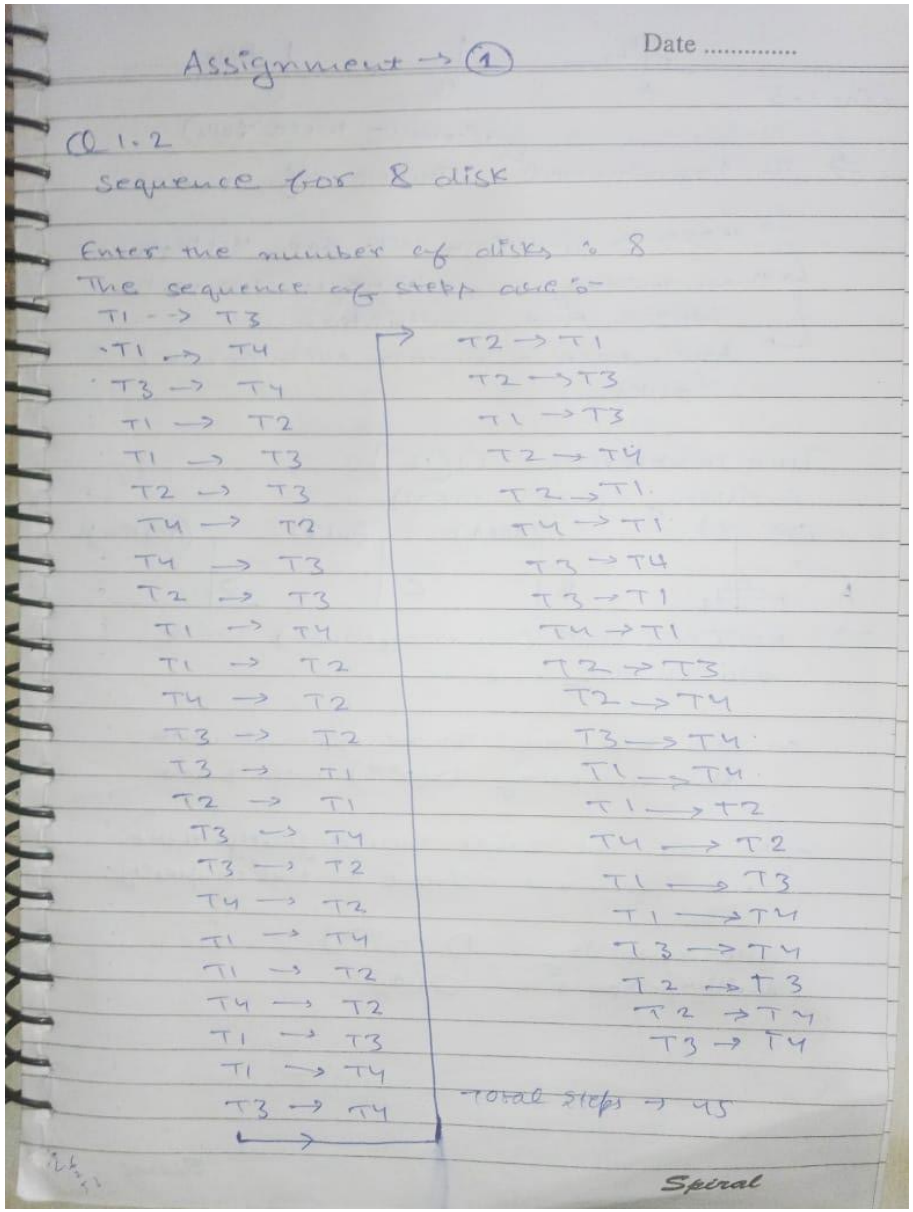
At Toh(1, '1', '3', '4', '2'), it will print the movement of the smaller disk from peg 1 to peg 3 .

Then, the repetition will go backwards. Apx. the remaining information in the previous stack frame.

Print the movement of the main wheel depending on the position of the accessory peg. This process continues until all the disks are moved from peg 1 to peg 4 .

Output will show the sequence of moves required to solve the Tower of Hanoi puzzle for disc 8. Each line represents the movement of the wheel from one peg to the next

Question 1.2



Assignment → (1) Date .....

Q 1.2

Sequence for 8 disk

Enter the number of disks : 8

The sequence of steps are :-

T1 → T3
T1 → T4
T3 → T4
T1 → T2
T1 → T3
T2 → T3
T4 → T2
T4 → T3
T2 → T3
T1 → T4
T1 → T2
T4 → T2
T3 → T2
T3 → T1
T2 → T1
T3 → T4
T3 → T2
T4 → T2
T1 → T4
T1 → T2
T4 → T2
T1 → T3
T1 → T4
T3 → T4

T2 → T1
T2 → T3
T1 → T3
T2 → T4
T2 → T1
T4 → T1
T3 → T4
T3 → T1
T4 → T1
T2 → T3
T2 → T4
T3 → T4
T1 → T4
T1 → T2
T4 → T2
T1 → T3
T1 → T4
T3 → T4
T2 → T3
T2 → T4
T3 → T4

total steps → 45

Spiral

Question 2.2

Date .....

Q 2.2 ⟹ eg→

Input array [32, 42, 1, 4, 32, 15, 6]

⟹ Initialisation

↳ the stack initialisat^n each as a sorted subarray of size one.

⟹ Merging iteratively

↳ ~~iteration~~ recursive merge pair until only one sorted subarray remains in stack hold.

⟹ 2nd iterat^n

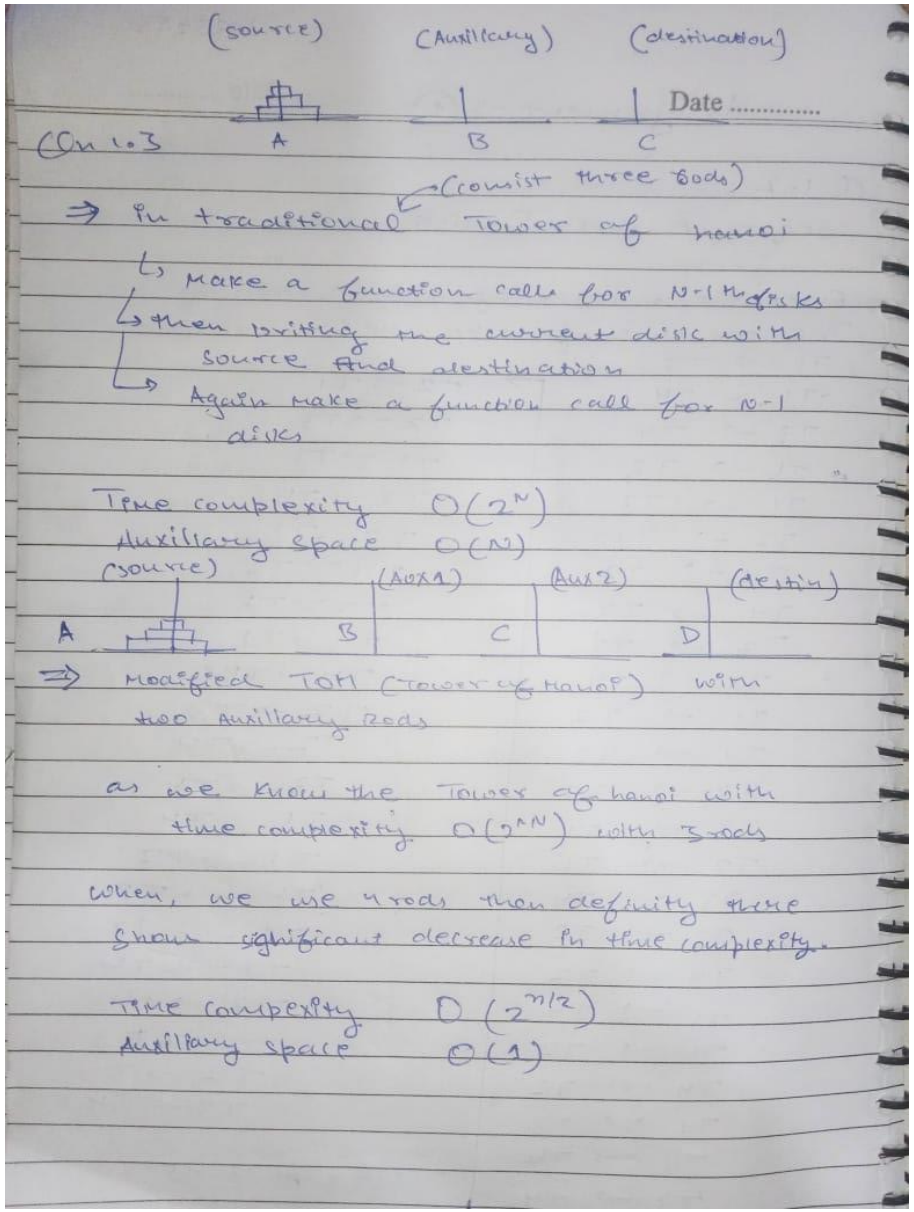
↳ POP [32, 42] & [1, 4] Merge Both to [1, 4, 32, 42] → Pushing Back to the stack

↳ POP [6, 15, 32] & [6, 15, 32] [only one remain] Merge them to [6, 6, 15, 15, 32, 32] → Pushing Back to stack

⟹ 3rd iteration

POP [1, 4, 32, 42] and [6, 6, 1515, 32, 32] Merge Both to [1, 4, 6, 6, 15, 15, 32, 32, 42]

Question 1.3


(source)
(Auxillary)
(destination)
A
B
C
Date .....
(Ch 1.3
(consist three rods)
⇒ In traditional Tower of hanoi
↳ make a function calls for N-1 disks
↳ then writing the current disk with source and destination
↳ Again make a function call for N-1 disks
Time complexity O(2^N)
Auxillary space O(N)
(source)
(Aux 1)
(Aux 2)
(destin)
A
B
C
D
⇒ Modified TOH (Tower of Hanoi) with two Auxillary rods
as we know the Tower of hanoi with time complexity O(2^N) with 3 rods
when, we use 4 rods then definity there shows significant decrease in time complexity.
Time complexity O(2^n/2)
Auxillary space O(1)