FACULTY NAME- PRAVIN KUMAR PANDEY
DEPARTMENT NAME- COMPUTER SCIENCE AND ENGINEERING
SUBJECT- DATA STRUCTURE USING- C
SEMESTER- 3
YEAR- 2

UNIT-2

Program for Tower of Hanoi

Tower of Hanoi is a mathematical puzzle where we have three rods and n disks. The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules-

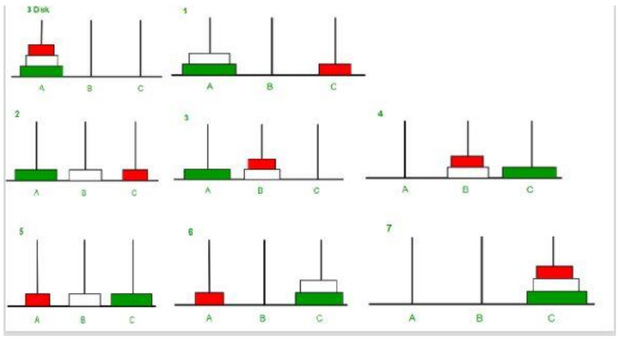
1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
3. No disk may be placed on top of a smaller disk.

Example: Let rod 1 = 'A', rod 2 = 'B', rod 3 = 'C'.

Step 1: Shift first disk from 'A' to 'B'.
Step 2: Shift second disk from 'A' to 'C'.
Step 3: Shift first disk from 'B' to 'C'.

The pattern here is-
Shift 'n-1' disks from 'A' to 'B'.
Shift last disk from 'A' to 'C'.
Shift 'n-1' disks from 'B' to 'C'.

Image illustration for 3 disks is-

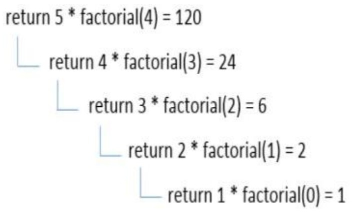


Recursion and Principles of Recursion

Recursion is the process which comes into existence when a function calls a copy of itself to work on a smaller problem. Any function which calls itself is called recursive function, and such function calls are called recursive calls. Recursion involves several numbers of recursive calls. However, it is important to impose a termination condition of recursion. Recursion code is shorter than iterative code however it is difficult to understand.
Recursion cannot be applied to all the problem, but it is more useful for the tasks that can be defined in terms of similar subtasks. For Example, recursion may be applied to sorting, searching, and traversal problems.
Generally, iterative solutions are more efficient than recursion since function call is always overhead. Any problem that can be solved recursively, can also be solved iteratively. However,

some problems are best suited to be solved by the recursion, for example, tower of Hanoi, Fibonacci series, factorial finding, etc.

Example:
```
int fact (int);
int main()
{
   int n,f;
   printf("Enter the number whose factorial you want to calculate?");
   scanf("%d",&n);
   f = fact(n);
   printf("factorial = %d",f);
}
int fact(int n)
{
   if (n==0)
   {
      return 0;
   }
   else if ( n == 1)
   {
      return 1;
   }
   else
   {
      return n*fact(n-1);
   }
}
```


return 5 * factorial(4) = 120
return 4 * factorial(3) = 24
return 3 * factorial(2) = 6
return 2 * factorial(1) = 2
return 1 * factorial(0) = 1

Tail Recursion
The tail recursion is basically using the recursive function as the last statement of the function. So, when nothing is left to do after coming back from the recursive call, that is called tail recursion.

Example
```
#include <iostream>
using namespace std;
void printN(int n){
   if(n < 0){
      return;
   }
   cout << n << " ";
   printN(n - 1);
}
int main() {
   printN(10);
}
```

References

[1]https://www.geeksforgeeks.org/c-program-for-tower-of-hanoi/
[2]https://www.tutorialspoint.com/tail-recursion-in-data-structures
[3]https://www.edutechlearners.com/download/books/DS.pdf
[4]https://www.iare.ac.in/sites/default/files/DS_NOTES_BY_Dr_L_V_NARASIMHA_PRASAD_0.pdf