Recursion

Topics

1. What is Recursion?
2. Problems
3. Solutions and Complexity
4. Important Notes

What is Recursion?

Recursion in computer science, is a method where the solution to a problem depends on solutions to smaller instances of the same problem (as opposed to iteration). The approach can be applied to many types of problems, and recursion is one of the central ideas of a computer.

- Most computer programming languages support recursion by allowing a function to call itself within the program text.
- A recursive function definition has one or more base cases, meaning input(s) for which the function produces a result trivially (without recurring).

Examples: Convert json string to object, expression evaluator with brackets etc.

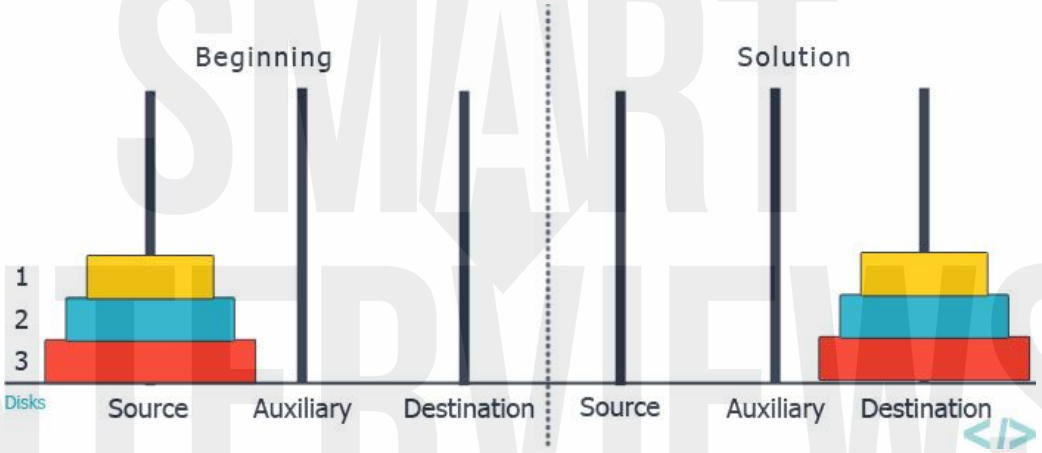
Problems

1. WAF to compute sum of numbers from 1 to n.

2. WAF to print Binary Representation of a given integer.

3. WAF to compute n^{th} Fibonacci number.

4. Towers of Hanoi – [**TOH @WolframAlpha**] - Given a stack of '*n*' disks arranged from largest on the bottom, to smallest on top, placed on a rod, together with two empty rods. The Towers of Hanoi puzzle asks us for the minimum number of moves required to move the stack from one rod to another, where moves are allowed only if they place smaller disks on top of larger disks.



5. WAF to print interleaving of 2 strings.

6. WAF to compute x^y.

Solutions and Complexity

1. WAF to compute sum of numbers from 1 to *n*.

```
int sum(int n) {
    if(n == 0)    return 0;
    return n + sum(n-1);
}
```

Recurrence: $T(n) = T(n-1) + 1$
Time Complexity: $O(n)$

2. WAF to print Binary Representation of a given integer.

```
void binaryRepresentation(int n) {
    if(n == 0)    return;
    binaryRepresentation(n/2);
    print n%2;
}
```

Recurrence: $T(n) = T(n/2) + 1$
Time Complexity: $O(\log_2(n))$

3. WAF to compute n^{th} Fibonacci number.

```
int fibo(int n) {
    if(n <= 2)    return 1;
    return fibo(n-1) + fibo(n-2);
}
```

Recurrence: $T(n) = T(n-1) + T(n-2) + 1$
Time Complexity: $O(2^n)$ [**Complexity Analysis of Fibonacci Series @YouTube**]

4. Towers of Hanoi [**TOH Simulation in 3D**]

```
void TOH(int n, char src, char dest, char temp) {
    if(n == 0)    return;
    TOH(n-1, src, temp, dest);
    printf("Move %d from %c to %c\n", n, src, dest);
    TOH(n-1, temp, dest, src);
}
```

Recurrence: $T(n) = 2T(n-1) + 1$
Time Complexity: $O(2^n)$

5. WAF to print interleaving of 2 strings.

```
void interleavings(char *A, char *B, char *ans, int m, int n, int idx) {
    if(m==0 && n==0) {
        print ans;
        return ;
    }
    if(m != 0) {
        ans[idx] = A[0];
        interleavings(A+1, B, ans, m-1, n, idx+1);
    }
    if(n != 0) {
        ans[idx] = B[0];
        interleavings(A, B+1, ans, m, n-1, idx+1);
    }
}
```

Number of Interleavings: `count(m, n) = count(m-1, n) + count(m, n-1)`
`count(1, 0) = 1` and `count(0, 1) = 1`

6. WAF to compute x^y.

```
int power(int x, unsigned int y) {
    if(y == 0)
        return 1;

    int z = power(x, y/2);
    if(y % 2 == 0)
        return z*z;
    else
        return x*z*z;
}
```

Recurrence: $T(n) = T(n/2) + 1$
Time Complexity: $O(\log_2(n))$

Important Notes

Fibonacci Numbers:
The fibonacci numbers/sequence are numbers in the following integer sequence:
`1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...`
or (often, in modern usage): `0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...`

By definition, the first two numbers in the fibonacci sequence are either `1` and `1`, or `0` and `1`, depending on the chosen starting point of the sequence, each subsequent number is the sum of the previous two.
In mathematical terms, the sequence F_n of fibonacci numbers is defined by the recurrence relation: $F_n = F_{n-1} + F_{n-2}$, with values initialized as: $F_1 = 1, F_2 = 1$; or, $F_1 = 0, F_2 = 1$;

Interleavings:
An interleaved string of given two strings preserves the order of characters in individual strings.
Example:
1. `str1 = "AB", str2 = "CD", Output: ABCD, ACBD, ACDB, CABD, CADB, CDAB`
2. `str1 = "AB", str2 = "C", Output: ABC, ACB, CAB`