Tower of Hanoi

Josemar Rocha da Silva
Professor: Herbert Oliveira Rocha

A bendita torre

Algoritmo

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

void hanoi(int n, char ponto_atual, char destino, char auxiliar){
    if(n==1){
        printf("\n Movendo disco 1 da coluna %c para %c", ponto_atual, destino);
        return;
    }
    hanoi(n-1, ponto_atual, auxiliar, destino);
    printf("\n Movendo disco %d da coluna %c para %c", n , ponto_atual, destino);
    hanoi(n-1, auxiliar, destino, ponto_atual);
}


int main()
{
    int n = 3; //número de discos
    hanoi(n, 'A', 'C', 'B');
    return 0;
}
```

Experimentos com o algoritmo

```
C:\Users\Pichau\Desktop\codes\hanoi\bin\Debug\hanoi.exe

 Movendo disco 1 da coluna A para B
 Movendo disco 2 da coluna A para C
 Movendo disco 1 da coluna B para C
Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```

Número de discos: 2

Número de movimentos: 3

```
C:\Users\Pichau\Desktop\codes\hanoi\bin\Debug\hanoi.exe

 Movendo disco 1 da coluna A para C
 Movendo disco 2 da coluna A para B
 Movendo disco 1 da coluna C para B
 Movendo disco 3 da coluna A para C
 Movendo disco 1 da coluna B para A
 Movendo disco 2 da coluna B para C
 Movendo disco 1 da coluna A para C
Process returned 0 (0x0)   execution time : 0.020 s
Press any key to continue.
```

Número de discos: 3

Número de movimentos: 7

Assim conseguimos analisar que o número de movimentos e o tempo que o algoritmo leva para terminar está ligado ao número de discos, logo, quanto mais discos, mais movimentos, esses que podem ser encontrados usando: $(2^n)-1$.

Ex: n = 40, o número de movimentos para finalizar é 1.099.511.627.775.

Definindo

```
T(1) = 1

T(n) = 2 * T(n-1) + 1, se n >= 2.
```

n	T(n)
1	1
2	3
3	7
4	15

Recorrência 

Como pôde ser visto no slide anterior de experimentos, vimos que o algoritmo termina ao finalizarmos os movimentos, que são (2^n)-1, onde n é o número de discos. Agora devemos provar isso.

T(n) = 2*T(n-1) + 1

T(n) = 2 * (2 * T(n-2) + 1) + 1

T(n) = (2^2) * T(n-2) + 2^1 + 2^0

T(n) = (2^k) * T(n-k) + 2^(k-1) + 2^(k-2) + ... + 2^0 ; Com **n = k** e **n - k = 0**

T(n) = (2^n) * T(0) + 2^(n-1) + 2^(n-2) + … + 2^1 + 2^0 ; Esta é uma progressão geométrica.

T(n) = (2^n) - 1

Fatos interessantes sobre a torre

- A torre de hanoi é utilizada em esquemas de rotação de backup de dados.

- Surgiu como um brinquedo.

Ze End