

Bài 1: Hàm đệ quy tính tổng các số từ 1 đến n

****Giải thích từng bước khi $n = 7$ **:**

1. Gọi hàm `sum_to_n(7)`.
 - $n = 7$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 7 + sum_to_n(6)`.
2. Gọi hàm `sum_to_n(6)`.
 - $n = 6$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 6 + sum_to_n(5)`.
3. Gọi hàm `sum_to_n(5)`.
 - $n = 5$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 5 + sum_to_n(4)`.
4. Gọi hàm `sum_to_n(4)`.
 - $n = 4$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 4 + sum_to_n(3)`.
5. Gọi hàm `sum_to_n(3)`.
 - $n = 3$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 3 + sum_to_n(2)`.
6. Gọi hàm `sum_to_n(2)`.
 - $n = 2$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 2 + sum_to_n(1)`.
7. Gọi hàm `sum_to_n(1)`.
 - $n = 1$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 1 + sum_to_n(0)`.
8. Gọi hàm `sum_to_n(0)`.
 - $n = 0$, thỏa mãn điều kiện cơ sở.
 - Hàm trả về 0.

Các hàm được thực hiện lần lượt từ dưới lên trên:

- ``sum_to_n(1)`` trả về ``1 + 0 = 1``
- ``sum_to_n(2)`` trả về ``2 + 1 = 3``
- ``sum_to_n(3)`` trả về ``3 + 3 = 6``
- ``sum_to_n(4)`` trả về ``4 + 6 = 10``
- ``sum_to_n(5)`` trả về ``5 + 10 = 15``
- ``sum_to_n(6)`` trả về ``6 + 15 = 21``
- ``sum_to_n(7)`` trả về ``7 + 21 = 28``

Kết quả cuối cùng là 28.

Bài 2: Hàm đệ quy tính số Fibonacci thứ n

****Giải thích từng bước khi $n = 8$ **:**

1. Gọi hàm ``fibonacci(8)``.
 - ``n = 8``, không thỏa mãn điều kiện cơ sở.
 - Thực hiện ``return fibonacci(7) + fibonacci(6)``.
2. Gọi hàm ``fibonacci(7)``.
 - ``n = 7``, không thỏa mãn điều kiện cơ sở.
 - Thực hiện ``return fibonacci(6) + fibonacci(5)``.
3. Gọi hàm ``fibonacci(6)``.
 - ``n = 6``, không thỏa mãn điều kiện cơ sở.
 - Thực hiện ``return fibonacci(5) + fibonacci(4)``.
4. Gọi hàm ``fibonacci(5)``.
 - ``n = 5``, không thỏa mãn điều kiện cơ sở.
 - Thực hiện ``return fibonacci(4) + fibonacci(3)``.
5. Gọi hàm ``fibonacci(4)``.

- $n = 4$, không thỏa mãn điều kiện cơ sở.
- Thực hiện `return fibonacci(3) + fibonacci(2)`.

6. Gọi hàm `fibonacci(3)`.

- $n = 3$, không thỏa mãn điều kiện cơ sở.
- Thực hiện `return fibonacci(2) + fibonacci(1)`.

7. Gọi hàm `fibonacci(2)`.

- $n = 2$, không thỏa mãn điều kiện cơ sở.
- Thực hiện `return fibonacci(1) + fibonacci(0)`.

8. Gọi hàm `fibonacci(1)`.

- $n = 1$, thỏa mãn điều kiện cơ sở.
- Hàm trả về 1.

9. Gọi hàm `fibonacci(0)`.

- $n = 0$, thỏa mãn điều kiện cơ sở.
- Hàm trả về 0.

Các hàm được thực hiện lần lượt từ dưới lên trên:

- `fibonacci(2)` trả về $1 + 0 = 1$
- `fibonacci(3)` trả về $1 + 1 = 2$
- `fibonacci(4)` trả về $2 + 1 = 3$
- `fibonacci(5)` trả về $3 + 2 = 5$
- `fibonacci(6)` trả về $5 + 3 = 8$
- `fibonacci(7)` trả về $8 + 5 = 13$
- `fibonacci(8)` trả về $13 + 8 = 21$

Kết quả cuối cùng là 21.

Bài 3: Hàm đệ quy tính x mũ n

****Giải thích từng bước khi $x = 2$ và $n = 6$ **:**

1. Gọi hàm `power(2, 6)`.
 - $n = 6$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 2 * power(2, 5)`.
2. Gọi hàm `power(2, 5)`.
 - $n = 5$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 2 * power(2, 4)`.
3. Gọi hàm `power(2, 4)`.
 - $n = 4$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 2 * power(2, 3)`.
4. Gọi hàm `power(2, 3)`.
 - $n = 3$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 2 * power(2, 2)`.
5. Gọi hàm `power(2, 2)`.
 - $n = 2$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 2 * power(2, 1)`.
6. Gọi hàm `power(2, 1)`.
 - $n = 1$, không thỏa mãn điều kiện cơ sở.
 - Thực hiện `return 2 * power(2, 0)`.
7. Gọi hàm `power(2, 0)`.
 - $n = 0$, thỏa mãn điều kiện cơ sở.
 - Hàm trả về 1.

Các hàm được thực hiện lần lượt từ dưới lên trên:

- `power(2, 1)` trả về $2 * 1 = 2$
- `power(2, 2)` trả về $2 * 2 = 4$
- `power(2, 3)` trả về $2 * 4 = 8$
- `power(2, 4)` trả về $2 * 8 = 16$

- `power(2, 5)` trả về $2 * 16 = 32$

- `power(2, 6)` trả về $2 * 32 = 64$

Kết quả cuối cùng là 64.

Bài 4: Hàm đệ quy giải bài toán Tháp Hà Nội

****Giải thích từng bước khi $n = 4$ **:**

1. Gọi hàm `hanoi(4, 'A', 'B', 'C')`.

- $n = 4$, không thỏa mãn điều kiện cơ sở.

- Thực hiện `hanoi(3, 'A', 'C', 'B')`, in "Move disk 4 from A to B", sau đó `hanoi(3, 'C', 'B', 'A')`.

2.

Gọi hàm `hanoi(3, 'A', 'C', 'B')`.

- $n = 3$, không thỏa mãn điều kiện cơ sở.

- Thực hiện `hanoi(2, 'A', 'B', 'C')`, in "Move disk 3 from A to C", sau đó `hanoi(2, 'B', 'C', 'A')`.

3. Gọi hàm `hanoi(2, 'A', 'B', 'C')`.

- $n = 2$, không thỏa mãn điều kiện cơ sở.

- Thực hiện `hanoi(1, 'A', 'C', 'B')`, in "Move disk 2 from A to B", sau đó `hanoi(1, 'C', 'B', 'A')`.

4. Gọi hàm `hanoi(1, 'A', 'C', 'B')`.

- $n = 1$, thỏa mãn điều kiện cơ sở.

- In "Move disk 1 from A to C".

5. Gọi hàm `hanoi(1, 'C', 'B', 'A')`.

- $n = 1$, thỏa mãn điều kiện cơ sở.

- In "Move disk 1 from C to B".

...

Các bước tiếp tục như trên cho đến khi toàn bộ 4 đĩa được chuyển từ cọc A sang cọc B qua cọc C.

Bài 5: Hàm đệ quy giải bài toán cổ vừa gà vừa chó

****Giải thích từng bước**:**

1. Gọi hàm `ga_va_cho(H, L)`.

- Nếu `L` không chia hết cho 2 hoặc `L < 2 * H` hoặc `L > 4 * H`, thì không có giải pháp hợp lệ.

- Nếu `L == 2 * H`, thì tất cả đều là gà.

- Nếu `L == 4 * H`, thì tất cả đều là chó.

- Nếu không, tính số lượng chó `cho = (L - 2 * H) // 2`, số lượng gà `ga = H - cho`.

Ví dụ: Với $H = 10$ và $L = 32$

- $L \% 2 == 0$

- $L \geq 2 * H$ và $L \leq 4 * H$

- $cho = (32 - 2 * 10) // 2 = 6$

- $ga = 10 - 6 = 4$

Kết quả là 4 gà và 6 chó.

EM GỬI BÀI TẬP CÔ GIÁO SIÊU ĐÁNG YÊU NHÉ !!!!!!!