Nguyễn Văn Long_DHKL17A2HN_23174600035_CA CHIỀU

Bài 1:

Bước 1: hàm sum_of_numbers(7) được gọi: Hàm kiểm tra điều kiện n == 1. Vì 7 != 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về 7 + sum_of_numbers(6).

Bước 2:hàm sum_of_numbers(6) được gọi: Hàm kiểm tra điều kiện n == 1. Vì 6 != 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về 6 + sum_of_numbers(5). Tiếp

Bước 3: hàm sum_of_numbers(5) được gọi: Hàm kiểm tra điều kiện n == 1. Vì 5 != 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về 5 + sum_of_numbers(4).

Bước 4: hàm sum_of_numbers(4) được gọi: Hàm kiểm tra điều kiện n == 1. Vì 4 != 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về 4 + sum_of_numbers(3).

Bước 5: hàm sum_of_numbers(3) được gọi: Hàm kiểm tra điều kiện n == 1. Vì 3 != 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về 3 + sum_of_numbers(2).

Bước 6: hàm sum_of_numbers(2) được gọi: Hàm kiểm tra điều kiện n == 1. Vì 2 != 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về 2 + sum_of_numbers(1).

Bước 7:hàm sum_of_numbers(1) được gọi: Hàm kiểm tra điều kiện n == 1. Vì 1 == 1, nó sẽ trả về 1. Bây giờ, các kết quả được trả về từ các lần gọi đệ quy sẽ được tính toán lại: sum_of_numbers(1) = 1 sum_of_numbers(2) = 2 + 1 = 3 sum_of_numbers(3) = 3 + 3 = 6 sum_of_numbers(4) = 4 + 6 = 10 sum_of_numbers(5) = 5 + 10 = 15 sum_of_numbers(6) = 6 + 15 = 21 sum_of_numbers(7) = 7 + 21 = 28 Vì vậy, kết quả cuối cùng của sum_of_numbers(7) là 28.

....

Bài 2:

Bước 1: hàm fibonacci(8) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 8 > 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về fibonacci(7) + fibonacci(6).

Bước 2:hàm fibonacci(7) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 7 > 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về fibonacci(6) + fibonacci(5).

Bước 3: hàm fibonacci(6) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 6 > 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về fibonacci(5) + fibonacci(4).

Bước 4: hàm fibonacci(5) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 5 > 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về fibonacci(4) + fibonacci(3).

Bước 5: hàm fibonacci(4) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 4 > 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về fibonacci(3) + fibonacci(2).

Bước 6:hàm fibonacci(3) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 3 > 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về fibonacci(2) + fibonacci(1). Tiếp tục, hàm fibonacci(2) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 2 > 1, nó sẽ thực hiện else block. Trong else block, hàm sẽ trả về fibonacci(1) + fibonacci(0).

Bước 7:hàm fibonacci(1) và fibonacci(0) được gọi: Hàm kiểm tra điều kiện n <= 1. Vì 1 <= 1, nó sẽ trả về 1. Hàm kiểm tra điều kiện n <= 1. Vì 0 <= 1, nó sẽ trả về 0.

Bước 8:các kết quả được trả về từ các lần gọi đệ quy sẽ được tính toán lại: fibonacci(0) = 0 fibonacci(1) = 1 fibonacci(2) = fibonacci(1) + fibonacci(0) = 1 + 0 = 1 fibonacci(3) = fibonacci(2) + fibonacci(1) = 1 + 1 = 2 fibonacci(4) = fibonacci(3) + fibonacci(2) = 2 + 1 = 3 fibonacci(5) = fibonacci(4) + fibonacci(3) = 3 + 2 = 5 fibonacci(6) = fibonacci(5) + fibonacci(4) = 5 + 3 = 8 fibonacci(7) = fibonacci(6) + fibonacci(5) = 8 + 5 = 13 fibonacci(8) = fibonacci(7) + fibonacci(6) = 13 + 8 = 21 Vì vậy, kết quả cuối cùng của fibonacci(8) là 21.

....

Bài 3:

Buowsc1: định nghĩa hàm power(x, n) với hai tham số x và n.

Bước 2: kiểm tra điều kiện dừng: Nếu n == 0, trả về 1. Điều này là do bất kỳ số nào mũ 0 đều bằng 1.

Bước 3:nếu n != 0, thực hiện phép tính x * power(x, n-1).

Bước 4: gọi hàm power(2, 6) để tính 2**6 . Cụ thể, khi và , quá trình thực hiện sẽ như sau:

Gọi power(2, 6).

Kiểm tra n == 6 != 0, nên thực hiện 2 * power(2, 6-1).

Gọi power(2, 5).

Kiểm tra n == 5 != 0, nên thực hiện 2 * power(2, 5-1).

Gọi power(2, 4).

Kiểm tra n == 4 != 0, nên thực hiện 2 * power(2, 4-1).

Gọi power(2, 3).

Kiểm tra n == 3 != 0, nên thực hiện 2 * power(2, 3-1).

Gọi power(2, 2).

Kiểm tra n == 2 != 0, nên thực hiện 2 * power(2, 2-1).

Gọi power(2, 1).

Kiểm tra n == 1 != 0, nên thực hiện 2 * power(2, 1-1).

Gọi power(2, 0).

Kiểm tra n == 0, trả về 1.

Quay ngược lại các bước, tính toán kết quả: power(2, 1) = 2 * 1 = 2 power(2, 2) = 2 * 2 = 4 power(2, 3) = 2 * 4 = 8 power(2, 4) = 2 * 8 = 16 power(2, 5) = 2 * 16 = 32 power(2, 6) = 2 * 32 = 64 Vì vậy, kết quả của power(2, 6) là 64.

....

Bài 4:

Bước 1:Khi gọi thap_ha_noi(4, "A", "C", "B"), hàm sẽ thực hiện các bước sau: Với n = 4 (số đĩa), hàm sẽ không thỏa mãn điều kiện n == 1, nên sẽ thực hiện các bước sau:

1. Gọi đệ quy thap_ha_noi(4-1, "A", "B", "C"), tức là chuyển 3 đĩa từ cột A sang cột C, với cột B làm trung gian.

2. In ra thông báo "Chuyển đĩa 4 từ cột A sang cột B".

3. Gọi đệ quy thap_ha_noi(4-1, "C", "A", "B"), tức là chuyển 3 đĩa từ cột C sang cột B, với cột A làm trung gian.

Bước 2:Khi gọi thap_ha_noi(3, "A", "B", "C") (lần đầu tiên), hàm sẽ thực hiện các bước sau:. Với n = 3, hàm sẽ không thỏa mãn điều kiện n == 1, nên sẽ thực hiện các bước sau:

1. Gọi đệ quy thap_ha_noi(3-1, "A", "C", "B"), tức là chuyển 2 đĩa từ cột A sang cột C, với cột B làm trung gian.

2. In ra thông báo "Chuyển đĩa 3 từ cột A sang cột B"

3. Gọi đệ quy thap_ha_noi(3-1, "C", "A", "B"), tức là chuyển 2 đĩa từ cột C sang cột B, với cột A làm trung gian.

Bước 3: Tiếp tục như vậy, hàm sẽ gọi đệ quy cho đến khi n == 1, lúc này hàm sẽ in ra thông báo "Chuyển đĩa 1 từ cột A sang cột B" và kết thúc.

....

Bài 5:

Bước 1:Định nghĩa hàm cho_ga với 2 tham số đầu vào là tong_so_con (tổng số con) và tong_so_chan (tổng số chân).

Bước 2: Kiểm tra điều kiện cơ sở: Nếu tong_so_con và tong_so_chan đều bằng 0, nghĩa là không có con gà và con chó nào, hàm trả về 0, 0. Nếu tong_so_chan chia 2 dư 1, nghĩa là tổng số chân không thể chia đều cho gà và chó, hàm trả về -1, -1 để báo lỗi.

Bước 3: Sử dụng vòng lặp for để duyệt qua các giá trị của số chó cho từ 0 đến tong_so_con.

Tính số gà ga bằng cách lấy tong_so_con trừ đi số chó .

Kiểm tra xem tổng số chân của gà và chó có bằng tong_so_chan hay không.

Nếu bằng, hàm trả về số chó cho và số gà ga.

Nếu không, gọi đệ quy hàm cho_ga với tong_so_con - 1 và tong_so_chan - 4 (vì mỗi con chó có 4 chân).

Nếu kết quả trả về là -1, -1, tăng số chó lên 1 và trả về.

Nếu không, tiếp tục vòng lặp.

Bước 4: Nếu không tìm được kết quả, hàm trả về -1, -1 để báo lỗi.

Bước 5: chúng ta khởi tạo tong_so_chan là 100 và tong_so_con là 36, sau đó gọi hàm cho_ga với hai tham số này và in ra kết quả.

Quy trình đệ quy giúp chúng ta dần dần tìm ra số gà và số chó thỏa mãn điều kiện bài toán. Mỗi lần gọi đệ quy, chúng ta giảm số con và số chân đi để tìm ra kết quả.