Trần Thị Thanh Tâm_DHKL17A2HN_23174600095_CA2

Câu 1:

1. def sum_of_numbers(n):

   - Định nghĩa hàm sum_of_numbers với tham số n.

2. If n == 1:

   - Kiểm tra điều kiện cơ sở của hàm đệ quy: nếu n bằng 1, thì trả về 1.

3. return 1

   - Nếu n bằng 1, hàm trả về giá trị 1.

4. else:

   - Nếu n không bằng 1, thực hiện bước tiếp theo.

5. return n + sum_of_numbers(n-1)

   - Tính toán giá trị trả về bằng cách cộng n và gọi đệ quy sum_of_numbers(n-1).

   - Khi n = 7, sum_of_numbers(7) sẽ trả về 7 + sum_of_numbers(6).

Bây giờ, hãy xem xét chuỗi gọi đệ quy này khi n = 7:

1. sum_of_numbers(7)

   -7 + sum_of_numbers(6)

2. sum_of_numbers(6)

   - 6 + sum_of_numbers(5)

3. sum_of_numbers(5)

   - 5 + sum_of_numbers(4)

4. sum_of_numbers(4)

   - 4 + sum_of_numbers(3)

5. sum_of_numbers(3)

   - 3 + sum_of_numbers(2)

6. sum_of_numbers(2)

   - 2 + sum_of_numbers(1)

7. sum_of_numbers(1)

   - Giá trị cơ sở: 1

Trần Thị Thanh Tâm_DHKL17A2HN_23174600095_CA2

Sau khi đi qua các bước đệ quy, kết quả của sum_of_numbers(7) là:

7 + 6 + 5 + 4 + 3 + 2 + 1 = 28

Vì vậy, khi gọi print(sum_of_numbers(7)), kết quả đầu ra sẽ là `28`.

Câu 2:

Bước 1. fibonacci(8) được gọi.

Bước 2. Vì 8 > 1, nên hàm sẽ gọi đệ quy fibonacci(7) và fibonacci(6).

Bước 3. fibonacci(7) được gọi.

- Vì 7 > 1, nên hàm sẽ gọi đệ quy fibonacci(6) và fibonacci(5).
- fibonacci(6) được gọi.
  - Vì 6 > 1, nên hàm sẽ gọi đệ quy fibonacci(5) và fibonacci(4).
  - fibonacci(5) được gọi.
    - Vì 5 > 1, nên hàm sẽ gọi đệ quy fibonacci(4) và fibonacci(3).
    - fibonacci(4) được gọi.
      - Vì 4 > 1, nên hàm sẽ gọi đệ quy fibonacci(3) và fibonacci(2).
      - fibonacci(3) được gọi.
        - Vì 3 > 1, nên hàm sẽ gọi đệ quy fibonacci(2) và fibonacci(1).
        - `fibonacci(2) được gọi.
          - Vì 2 > 1, nên hàm sẽ gọi đệ quy fibonacci(1) và fibonacci(0).
          - fibonacci(1) được gọi và trả về 1.
          - fibonacci(0) được gọi và trả về 0.
          - fibonacci(2) trả về fibonacci(1) + fibonacci(0) = 1 + 0 = 1.
        - fibonacci(1) trả về 1.
        - fibonacci(3) trả về fibonacci(2) + fibonacci(1) = 1 + 1 = 2.
      - fibonacci(2) trả về 1.
      - fibonacci(4) trả về fibonacci(3) + fibonacci(2) = 2 + 1 = 3.
    - fibonacci(3) trả về 2.
    - fibonacci(5) trả về fibonacci(4) + fibonacci(3) = 3 + 2 = 5.

- fibonacci(4) trả về 3.

- fibonacci(6) trả về fibonacci(5) + fibonacci(4) = 5 + 3 = 8.

- fibonacci(5) trả về 5.

- fibonacci(7) trả về fibonacci(6) + fibonacci(5) = 8 + 5 = 1.

Bước 4. fibonacci(6) trả về 8.

Bước 5. fibonacci(8) trả về fibonacci(7) + fibonacci(6) = 13 + 8 = 21.

Vì vậy, kết quả của fibonacci(8) là 21.

Câu 3:

Bước 1: Nếu n bằng 0, hàm sẽ trả về 1, vì bất kỳ số nào mũ 0 đều bằng 1.

Bước 2: Nếu n không bằng 0, hàm sẽ trả về x nhân với kết quả của hàm power(x, n-1). Đây chính là cơ chế đệ quy - hàm gọi chính nó với giá trị n-1 cho đến khi n bằng 0.

Khi thực hiện với x=2 và n=6, các bước diễn ra như sau:

Bước 3. power(2, 6) được gọi

n không bằng 0, nên return 2 * power(2, 6-1) được thực hiện

Bước 4. power(2, 5) được gọi

n không bằng 0, nên return 2 * power(2, 5-1) được thực hiện

Bước 5. power(2, 4) được gọi

n không bằng 0, nên return 2 * power(2, 4-1) được thực hiện

Bước 6. power(2, 3) được gọi

n không bằng 0, nên return 2 * power(2, 3-1) được thực hiện

Bước 7. power(2, 2) được gọi

n không bằng 0, nên return 2 * power(2, 2-1) được thực hiện

Bước 8. power(2, 1) được gọi

n không bằng 0, nên return 2 * power(2, 1-1) được thực hiện

Bước 9. power(2, 0) được gọi, n bằng 0, nên hàm trả về 1

Bước 10. Các bước từ 11-13 sẽ được đệ quy trở lại, và kết quả cuối cùng là 64.

Câu 4:

Trần Thị Thanh Tâm_DHKL17A2HN_23174600095_CA2

Bước 1: Gọi đầu tiên: thap_ha_noi(4, "A", "B", "C")

Chưa thoả mãn điều kiện cơ sở (n != 1).

Gọi đệ quy để chuyển 3 đĩa từ cột A sang cột C với cột B là trung gian: thap_ha_noi(3, "A", "C", "B")

Bước 2: Gọi đệ quy tiếp theo: thap_ha_noi(3, "A", "C", "B")

Chưa thoả mãn điều kiện cơ sở (n != 1).

Gọi đệ quy để chuyển 2 đĩa từ cột A sang cột B với cột C là trung gian: thap_ha_noi(2, "A", "B", "C")

Bước 3: Gọi đệ quy tiếp theo: thap_ha_noi(2, "A", "B", "C")

Chưa thoả mãn điều kiện cơ sở (n != 1).

Gọi đệ quy để chuyển 1 đĩa từ cột A sang cột C với cột B là trung gian: thap_ha_noi(1, "A", "C", "B")

Bước 4: Gọi đệ quy cơ sở: thap_ha_noi(1, "A", "C", "B")

Thoả mãn điều kiện cơ sở (n == 1).

In ra: Chuyển đĩa 1 từ cột A sang cột C.

Bước 5: Quay lại thap_ha_noi(2, "A", "B", "C")

In ra: Chuyển đĩa 2 từ cột A sang cột B.

Gọi đệ quy để chuyển 1 đĩa từ cột C sang cột B với cột A là trung gian: thap_ha_noi(1, "C", "B", "A")

Bước 6: Gọi đệ quy cơ sở: thap_ha_noi(1, "C", "B", "A")

Thoả mãn điều kiện cơ sở (n == 1).

In ra: Chuyển đĩa 1 từ cột C sang cột B.

Bước 7: Quay lại thap_ha_noi(3, "A", "C", "B")

In ra: Chuyển đĩa 3 từ cột A sang cột C.

Gọi đệ quy để chuyển 2 đĩa từ cột B sang cột C với cột A là trung gian: thap_ha_noi(2, "B", "C", "A")

Bước 8: Gọi đệ quy tiếp theo: thap_ha_noi(2, "B", "C", "A")

Chưa thoả mãn điều kiện cơ sở (n != 1).

Gọi đệ quy để chuyển 1 đĩa từ cột B sang cột A với cột C là trung gian:

thap_ha_noi(1, "B", "A", "C")

Bước 9: Gọi đệ quy cơ sở: thap_ha_noi(1, "B", "A", "C")

Thoả mãn điều kiện cơ sở (n == 1).

In ra: Chuyển đĩa 1 từ cột B sang cột A.

Bước 10: Quay lại thap_ha_noi(2, "B", "C", "A")

In ra: Chuyển đĩa 2 từ cột B sang cột C.

Gọi đệ quy để chuyển 1 đĩa từ cột A sang cột C với cột B là trung gian:

thap_ha_noi(1, "A", "C", "B")

Bước 11: Gọi đệ quy cơ sở: thap_ha_noi(1, "A", "C", "B")

Thoả mãn điều kiện cơ sở (n == 1).

In ra: Chuyển đĩa 1 từ cột A sang cột C.

Bước 12: Quay lại thap_ha_noi(4, "A", "B", "C")

In ra: Chuyển đĩa 4 từ cột A sang cột B.

Gọi đệ quy để chuyển 3 đĩa từ cột C sang cột B với cột A là trung gian:

thap_ha_noi(3, "C", "B", "A")

Bước 13: Gọi đệ quy tiếp theo: thap_ha_noi(3, "C", "B", "A")

Chưa thoả mãn điều kiện cơ sở (n != 1).

Gọi đệ quy để chuyển 2 đĩa từ cột C sang cột A với cột B là trung gian:

thap_ha_noi(2, "C", "A", "B")

Bước 14: Gọi đệ quy tiếp theo: thap_ha_noi(2, "C", "A", "B")

Chưa thoả mãn điều kiện cơ sở (n != 1).

Gọi đệ quy để chuyển 1 đĩa từ cột C sang cột B với cột A là trung gian:

thap_ha_noi(1, "C", "B", "A")

Bước 15: Gọi đệ quy cơ sở: thap_ha_noi(1, "C", "B", "A")

Thoả mãn điều kiện cơ sở (n == 1).

In ra: Chuyển đĩa 1 từ cột C sang cột B.

Bước 16: Quay lại thap_ha_noi(2, "C", "A", "B")

In ra: Chuyển đĩa 2 từ cột C sang cột A.

Gọi đệ quy để chuyển 1 đĩa từ cột B sang cột A với cột C là trung gian:

thap_ha_noi(1, "B", "A", "C")

Bước 17: Gọi đệ quy cơ sở: thap_ha_noi(1, "B", "A", "C")

Thoả mãn điều kiện cơ sở (n == 1).

In ra: Chuyển đĩa 1 từ cột B sang cột A.

Bước 18: Quay lại thap_ha_noi(3, "C", "B", "A")

In ra: Chuyển đĩa 3 từ cột C sang cột B.

Gọi đệ quy để chuyển 2 đĩa từ cột A sang cột B với cột C là trung gian:

thap_ha_noi(2, "A", "B", "C")

Bước 19: Gọi đệ quy tiếp theo: thap_ha_noi(2, "A", "B", "C")

Chưa thoả mãn điều kiện cơ sở (n != 1).

Gọi đệ quy để chuyển 1 đĩa từ cột A sang cột C với cột B là trung gian:

thap_ha_noi(1, "A", "C", "B")

Bước 20: Gọi đệ quy cơ sở: thap_ha_noi(1, "A", "C", "B")

Thoả mãn điều kiện cơ sở (n == 1).

In ra: Chuyển đĩa 1 từ cột A sang cột C.

Bước 21: Quay lại `thap_ha_noi(2, "A", "B", "C")

Câu 5:

Bước 1: Định nghĩa hàm cho_ga:

Tong_so_con: Tổng số con(chó + gà)

tong_so_chan: Tổng số chân của tất cả các con.

Bước 2: Nếu tổng số con và tổng số chân đều là 0, trả về 0, 0 (không có chó và không có gà).

Bước 3: Kiểm tra tổng số chân có phải số lẻ:

Nếu tổng số chân là số lẻ, không thể chia đều cho chó và gà (vì chó có 4 chân và gà có 2 chân).

Bước 4: Thử tất cả các giá trị có thể cho số chó:

Lặp qua tất cả các giá trị có thể của số chó (cho).

Tính số gà (ga) là tong_so_con - cho.

Nếu số chân của gà và chó khớp với tổng số chân (ga * 2 + cho * 4 == tong_so_chan), trả về số chó và số gà.

Bước 5: Gọi đệ quy để thử các giá trị khác:

Nếu không tìm được cách phân chia hợp lý trong vòng lặp, gọi đệ quy để thử các giá trị khác của tổng số con và tổng số chân (giảm số chó và số chân tương ứng).

Nếu tìm được cách phân chia hợp lý trong lần gọi đệ quy, trả về số chó và số gà đã tính toán.

Nếu không tìm được cách phân chia hợp lý trong lần gọi đệ quy, trả về -1, -1 (không hợp lý).

Bước 6: Gọi hàm và in kết quả:

Đặt tổng số con là 36 và tổng số chân là 100.

Gọi hàm cho_ga để tìm số chó và số gà.

In ra kết quả.

Với đầu vào tong_so_con = 36 và tong_so_chan = 100, hàm sẽ tính toán và đưa ra kết quả số chó và số gà phù hợp.