Bài 1.10: Thuật toán sinh tổ hợp chập k của n

- ✓ Mô tả bài toán
- ✓ Thuật toán tổng quát
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Mô tả bài toán

- Thuật toán sinh hoán tổ hợp chập k của n phần tử áp dụng thuật toán sinh cấu hình kế tiếp để giải quyết vấn đề.
- Bài toán: cho 2 số nguyên dương n, k với 1 <= k <= n. Viết chương trình sinh toàn bộ các tổ hợp chập k của n phần tử theo thứ tự từ điển.
- Cấu hình C = {c1, c2, …, ck} chứa k phần tử với ci = {1, 2, …, n} và i = {1, …, k} là một tổ hợp chập k của n phần tử.
- Nói cách khác, mỗi cấu hình tổ hợp chập k của n phần tử là bộ gồm k giá giá trị khác nhau của các số từ 1 đến n.
- Với một số nguyên dương n và k cho trước, ta có C_n^k cấu hình thỏa mãn.
- Ví dụ: tổ hợp chập 3 của 5 gồm 10 cấu hình sau: (1, 2, 3), (1, 2, 4), (1, 2, 5), (1, 3, 4), (1, 3, 5), (1, 4, 5), (2, 3, 4), (2, 3, 5), (2, 4, 5), (3, 4, 5).

Thuật toán tổng quát

➢ Sau đây là mã giả thuật toán sinh tổ hợp kế tiếp:

```
// thuật toán sinh tổ hợp kế tiếp
bool nextCombination(arr[], n) { // arr: chứa cấu hình hiện tại
    i = arr.length - 1; // xuất phát từ phần tử cuối của tổ hợp
    while(i >= 0 && arr[i] == n - k + i + 1) { // tìm phần tử arr[i] đầu tiên khác n-k+i+1
        i--;
    }
    if(i >= 0) { // nếu i chưa vượt quá phần tử trái cùng
        arr[i] = arr[i] + 1; // thay x[i] = x[i]+1
        for(j = i + 1; j < k; j++) { // cập nhật các phần tử từ vị trí i+1 đến k
            arr[j] = arr[i] + j - i; // gán arr[j] = arr[i] + j - i
        }
        return false; // thông báo cho nơi gọi biết đây chưa phải cấu hình cuối
    } else {
        return true; // thông báo cho nơi gọi biết đây là cấu hình cuối cùng
    }
}
// thuật toán sinh tổ hợp chập k của n
void generate(arr[], n) {
    isFinalConfig = false;
    while (!isFinalConfig)
    {
        output(arr);
        isFinalConfig = nextCombination(arr, n);
    }
}
```

Thuật toán tổng quát

➢ Sau đây là mã thật thuật toán sinh tổ hợp kế tiếp:

```
// thuật toán sinh tổ hợp kế tiếp
public static boolean nextCombination(int[] arr, int n) { // arr: chứa cấu hình hiện tại
    int i = arr.length - 1; // xuất phát từ phần tử cuối của tổ hợp
    int k = arr.length; // lấy số phần tử của mảng
    while (i >= 0 && arr[i] == n - k + i + 1) { // tìm phần tử arr[i] đầu tiên khác n-k+i+1
        i--; // giảm i
    }
    if (i >= 0) { // nếu i chưa vượt quá phần tử trái cùng
        arr[i] = arr[i] + 1; // thay x[i] = x[i]+1
        for (int j = i + 1; j < k; j++) { // cập nhật các phần tử từ vị trí i+1 đến k
            arr[j] = arr[i] + j - i; // gán arr[j] = arr[i] + j - i
        }
        return false; // thông báo cho nơi gọi biết đây chưa phải cấu hình cuối
    } else {
        return true; // thông báo cho nơi gọi biết đây là cấu hình cuối cùng
    }
}
```

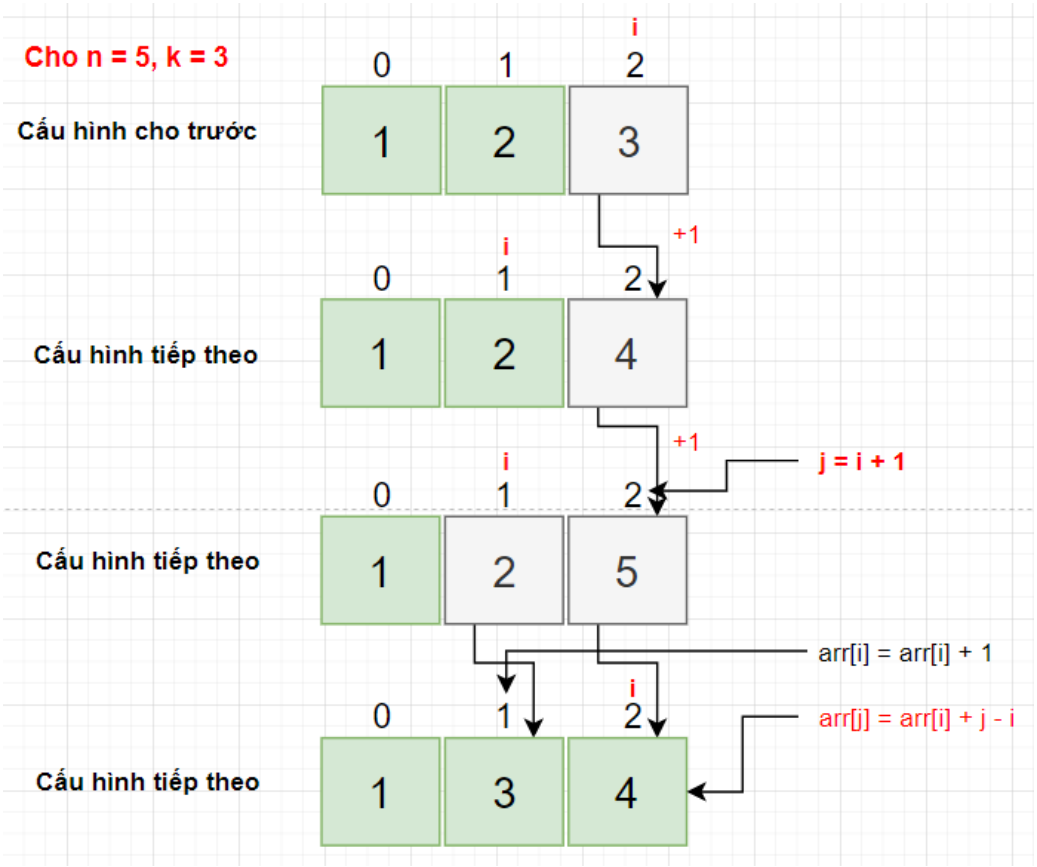
Thuật toán tổng quát

- Sau đây là mã thật thuật toán sinh tổ hợp chập k của n:

```
// thuật toán sinh tổ hợp chập k của n
public static void generate(int[] arr, int n) {
    boolean isFinalConfig = false; // biến đánh dấu cấu hình cuối
    while (!isFinalConfig) { // lặp chừng nào chưa đến cấu hình cuối
        output(arr); // hiển thị cấu hình hiện tại
        isFinalConfig = nextCombination(arr, n); // sinh cấu hình tiếp theo
    }
}
```

Ví dụ minh họa

➢ Ví dụ cho cấu hình 1 2 3, tìm 3 cấu hình kế tiếp:



Nội dung tiếp theo

Cấu trúc dữ liệu danh sách liên kết