Thuật toán này giúp liệt kê tất cả các xâu nhị phân có độ dài N, tương ứng với việc liệt kê tất cả các tập con của một tập N phần tử.

Với N = 3, ta có **2^N = 8** xâu nhị phân:

000

001

010

011

100

101

110

111

**1. Khai báo biến**

* n: Độ dài của xâu nhị phân.
* X[100]: Mảng lưu cấu hình của xâu nhị phân hiện tại.
* ok: Biến kiểm soát vòng lặp, xác định xem đã sinh xong tất cả xâu hay chưa.

**2. Khởi tạo cấu hình đầu tiên**

Cấu hình đầu tiên của xâu nhị phân là toàn bộ n số 0.

Ví dụ với n = 3:

000

Hàm ktao() thực hiện như sau:

void ktao(){

for(int i = 1; i <= n; i++){

X[i] = 0;

}

}

**3. Kiểm tra cấu hình cuối cùng**

Cấu hình cuối cùng là xâu toàn 1, ví dụ 111.

Hàm kiemTra() kiểm tra xem xâu hiện tại có phải xâu cuối cùng không:

bool kiemTra(){

for(int i = 1; i <= n; i++){

if(X[i] == 0) return false;

}

return true;

}

Nếu còn xuất hiện 0, nghĩa là chưa phải xâu cuối cùng.

**4. Sinh xâu nhị phân tiếp theo**

Nguyên tắc sinh:

* Duyệt từ cuối về đầu, tìm **bit 0 đầu tiên**.
* Đổi 0 đó thành 1, các bit sau nó đổi về 0.

**Ví dụ quá trình sinh với n = 3**:

Sao chépChỉnh sửa

000 → 001

001 → 010

010 → 011

011 → 100

100 → 101

101 → 110

110 → 111

Hàm sinh() thực hiện như sau:

void sinh(){

int i = n; // Duyệt từ cuối về đầu

while(i >= 1 && X[i] == 1){

X[i] = 0; // Nếu là bit 1 thì đổi thành 0

--i; // Lùi về trái

}

if(i == 0){

ok = false; // Nếu tất cả là 1, đây là cấu hình cuối cùng

}

else{

X[i] = 1; // Đổi bit đầu tiên tìm được từ 0 thành 1

}

}

**5. Vòng lặp sinh xâu nhị phân**

Sau khi khởi tạo xâu đầu tiên, chương trình liên tục sinh ra xâu tiếp theo cho đến khi đạt xâu cuối cùng.

Hàm main() thực hiện:

int main(){

cin >> n;

ktao(); // Khởi tạo xâu đầu tiên

while(ok){

for(int i = 1; i <= n; i++){

cout << X[i] << " ";

}

cout << endl;

sinh(); // Sinh xâu tiếp theo

}

}

Các bước thực hiện:

1. Nhập n.
2. Khởi tạo xâu đầu tiên (000...0).
3. In ra xâu hiện tại.
4. Gọi sinh() để tạo xâu tiếp theo.
5. Lặp lại cho đến khi không còn xâu mới (111...1).

**6. Độ phức tạp thuật toán**

* Có **2^N** xâu nhị phân cần sinh ra.
* Mỗi lần sinh mất **O(N)** thời gian.
* Tổng thời gian chạy: **O(N \* 2^N)**.

Thuật toán này rất hiệu quả cho N nhỏ nhưng sẽ chậm với N lớn (N > 20).

**7. Tổng kết**

* Thuật toán sử dụng mảng X để lưu xâu nhị phân hiện tại.
* Duyệt từ cuối về đầu để tìm bit 0 đầu tiên.
* Nếu tìm được, đổi nó thành 1, reset các bit sau thành 0.
* Dừng lại khi toàn bộ xâu là 1.
* Hiệu quả và dễ cài đặt để sinh tất cả xâu nhị phân có độ dài N.