Backtracking

1. 📚 Giới thiệu

Thuật toán **quay lui (backtracking)** là một kỹ thuật giải quyết bài toán bằng cách xây dựng lời giải từng bước và quay trở lại bước trước đó nếu hướng hiện tại không dẫn đến kết quả hợp lệ. Backtracking thường dùng cho các bài toán liệt kê, tìm kiếm lời giải thỏa mãn ràng buộc, hoặc tổ hợp.

🧠 Ý tưởng chính

1. Bắt đầu từ một trạng thái ban đầu.
2. Từng bước mở rộng lời giải bằng cách chọn lựa các phương án hợp lệ.
3. Nếu một bước dẫn đến vi phạm ràng buộc → quay lui và thử phương án khác.
4. Nếu đạt lời giải đầy đủ → lưu lại hoặc dừng theo yêu cầu bài toán

---

2. 🧩 Mã giả chung

```
void Try(i){
   Thử gán các giá trị có thể cho phần tử X[i]
   for (j = <Giá trị 1> ... <Giá trị m>){
         <Kiểm tra xem có gán được j cho X[i] hay không ?>
         X[i] = j;
         <Kiểm tra xem i có phải là phần tử cuối cùng của cấu hình không?>
         if (i == n){
            <Tìm được một cấu hình>
         }
         else {
            <Tiếp tuc đi xây dựng phần tử thứ i + 1>
            Try(i + 1);
         }
         <Backtrack>
         <Bỏ ghi nhận X[i] = j>
   }
   }
```

Tùy theo loại bài toán, cách sinh cấu hình tiếp theo sẽ thay đổi.

---

3. 🔢 Sinh xâu nhị phân độ dài n

✅ Mô tả:

Sinh tất cả các xâu nhị phân độ dài nnn: gồm các dãy chỉ chứa 0 và 1.

🔷 Code:

```
#include <iostream>
#include <vector>
using namespace std;
void Try(vector<int> a, int i){
```

```
    for (int j = 0; j <= 1; j++){
        a[i] = j;
        if (i == a.size() - 1){
            for (int x : a){
                cout << x << " ";
            }
            cout << endl;
        }
        else {
            Try(a, i + 1);
        }

    }
}
int main(){
    int n;
    cin >> n;
    vector<int> a(n);
    Try(a, 0);
}
```

🔶 Ví dụ (n = 3):

```
000
001
010
011
100
101
110
111
```

---

4. 🧱 Sinh tổ hợp chập K của N phần tử

✅ Mô tả:

Liệt kê tất cả các tổ hợp chập K của tập {1, 2, ..., N}. Mỗi tổ hợp là dãy tăng dần.

🔷 Code:

```
#include <iostream>
#include <vector>
using namespace std;
void Try(vector<int> a, int n, int k, int i = 1){
    for (int j = a[i - 1] + 1; j <= i + n - k; j++){
        a[i] = j;
        if (i == k){
            for (int i = 1; i <= k; i++){
                cout << a[i] << " ";
            }
```

```
            cout << endl;
        }
        else {
            Try(a, n, k, i + 1);
        }

    }
}
int main(){
    int n, k;
    cin >> n >> k;
    vector<int> a(k + 1);
    a[0] = 0;
    Try(a, n, k);
}
```

🔶 Ví dụ (N = 5, K = 3):

```
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
```

---

5. 🔁 Sinh hoán vị của N phần tử

✅ Mô tả:

Sinh tất cả các hoán vị của tập {1, 2, ..., N}. Mỗi hoán vị là dãy gồm N phần tử không trùng nhau.

🔷 Code:

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

// Khởi tạo cấu hình đầu tiên từ [1, 2, ..., n]
void init(vector<int>& a) {
    for (int i = 0; i < a.size(); i++) {
        a[i] = i + 1;
    }
}

// Hàm sinh hoán vị kế tiếp của dãy a theo thứ tự từ điển
```

```
void sinh(vector<int>& a, bool& final) {
    int i = a.size() - 2;

    // Tìm phần tử a[i] đầu tiên (từ phải sang trái) sao cho a[i] < a[i + 1]
    while (i >= 0 && a[i + 1] < a[i]) {
        i--;
    }

    // Nếu không tìm thấy (i == -1) → cấu hình hiện tại là hoán vị cuối cùng
    if (i == -1) {
        final = true;
    } else {
        // Tìm phần tử a[j] lớn nhất bên phải a[i] mà a[j] > a[i]
        int j = a.size() - 1;
        int temp = 0;
        while (j >= 0 && a[j] < a[i]){
            j--;
        }
        // Hoán đổi a[i] và a[j]
        swap(a[i], a[j]);
        // Đảo ngược đoạn từ a[i + 1] đến hết dãy để đảm bảo thứ tự tăng
        reverse(a.begin() + i + 1, a.end());
    }
}

int main() {
    int n;
    cin >> n;

    vector<int> a(n);
    init(a);  // Buoc 1: Khởi tạo cấu hình đầu tiên:  [1, 2, ..., n]

    bool final = false;  // Biến đánh dấu kết thúc sinh cấu hình

    // Buoc 2: Vòng lặp sinh tất cả các hoán vị
    while (!final) {
        // + Dua ra cấu hình hiện tại
        for (int x : a) {
            cout << x;
        }
        cout << endl;

        // Sinh cấu hình kế tiếp
        sinh(a, final);
    }
}
```

🔶 Ví dụ (N = 3):

```
1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1
```

✅ Tổng kết

Kỹ thuật	Đặc điểm chính	Số lượng cấu hình
Nhị phân	Duyệt tất cả dãy 0/1	2^n
Tổ hợp	Dãy tăng, không trùng	$\binom{n}{k}$
Hoán vị	Dãy không lặp, mọi vị trí khác nhau	n!