



ĐỘI TUYỂN OLP Tin học PTIT 2025

TUẦN 4: BÀI TOÁN INTERACTIVE



**Data Structure and
Algorithm**



Đặc điểm

- Bài toán lập trình thông thường: chương trình nhận input tĩnh của hệ thống, xử lý input và trả về output.
- Bài toán tương tác: chương trình tương tác với hệ thống – chương trình gửi đáp án hoặc yêu cầu cho hệ thống và nhận lại phản hồi dựa trên yêu cầu đã gửi.
- Input thực sự thường được ẩn đi và chỉ hệ thống biết.
- Output chính là yêu cầu của bài toán.
- Dữ liệu mà chương trình đọc có thể không giống nhau trên cùng một bộ test, tùy vào cách tương tác của chương trình đó.



Đặc điểm

- Cơ chế hoạt động cơ bản của hệ thống trong bài toán tương tác:
- Hệ thống gửi thông tin ban đầu (nếu có) và phản hồi cho chương trình vào trình tương tác (interactor), đồng thời nhận từ trình tương tác yêu cầu của chương trình.
- Chương trình gửi yêu cầu cho hệ thống vào trình tương tác, đồng thời nhận từ trình tương tác dữ liệu vào (nếu có) và phản hồi từ hệ thống.



Đặc điểm

- Chương trình cần đảm bảo đọc / ghi đúng và đủ các thông tin cần thiết.
- Về việc ghi thông tin, chương trình cần đảm bảo phải flush (đẩy) thông tin ra luồng stdout (vì có khả năng nó mới chỉ được lưu trữ tại bộ đệm).
- Giải pháp: sử dụng lệnh flush được thiết lập sẵn trong ngôn ngữ lập trình
 - C/C++: `fflush(stdout)` (nếu in bằng `printf`) / `std::cout << flush` (nếu in bằng `std::cout`).
 - Java: `System.out.flush()`
 - Python: `flush(output)`
- Note: `std::endl` có thể được dùng để flush trực tiếp ngay khi xuống dòng với C++.



Các lỗi có thể xảy ra

- Compilation error
- Wrong answer
- Time limit exceeded / Memory limit exceeded / Runtime error
- Idleness limit exceeded:
 - Xảy ra khi trình tương tác chờ quá lâu mà không nhận được yêu cầu.
 - Thường xảy ra do lỗi lập trình của thí sinh.
- Lỗi ngẫu nhiên:
 - Xảy ra khi trình tương tác đã đóng lại (do đã trả lời hoặc vượt giới hạn yêu cầu) nhưng chương trình vẫn chưa dừng lại.
 - Lỗi cụ thể xuất hiện xảy ra tùy vào chương trình.



Ví dụ 1: Tìm kiếm nhị phân

- Đoán một con số bí mật trên đoạn $[1, 1000000]$.
- Chương trình có thể gửi yêu cầu là một con số, và nhận về $<$ hoặc \geq nếu số đó nhỏ hơn số bí mật hoặc ngược lại.
- Tìm con số bí mật đó.
- Số lần gửi yêu cầu tối đa là 25.

- Ý tưởng: tìm kiếm nhị phân

<https://codeforces.com/gym/101021/problem/1>



Ví dụ 2: Số nguyên tố

- Có một con số bí mật trên đoạn $[2, 100]$.
 - Chương trình có thể gửi yêu cầu là một con số, và nhận về “yes” hoặc “no” nếu số đó là ước của số bí mật hoặc ngược lại.
 - Xác định số bí mật đó là số nguyên tố hay hợp số.
 - Số lần gửi yêu cầu tối đa là 20.
-
- Ý tưởng: giải thuật kiểm tra tính nguyên tố của một số

<https://codeforces.com/problemset/problem/679/A>



Ví dụ 3: Truy vấn trên cây

- Cho một cây có N đỉnh
 - Mỗi truy vấn “? u ” được phép hỏi khoảng cách từ 1 đỉnh u tới $N-1$ đỉnh còn lại
 - Xác định cấu hình của cây
 - Số lần gửi yêu cầu tối đa là $\lceil n/2 \rceil$.
-
- Ý tưởng: phân cấp đồ thị thành 2 phần: khoảng cách chẵn và lẻ tới đỉnh gốc 1.

<https://codeforces.com/contest/1534/problem/D>



Viết trình chấm cho bài toán Interactive

- Phải tự sinh test case với lời giải của mình (gán đáp án sẵn, rồi in ra truy vấn)
- Viết trình chấm và so sánh kết quả khi chạy bằng trình chấm và lời giải của mình
- Đặt/Xóa define để tắt/bật grader
- Tạo 1 struct grader:
 - Init() → khởi tạo đáp án
 - Ask() → in ra truy vấn và trả về câu trả lời cho truy vấn
 - Yes() → In ra đáp án cuối cùng
- Trong hàm Ask phải viết assert để thỏa mãn các điều kiện của đề bài



Viết trình chấm cho bài toán Interactive

- Bài toán 1: Chặt nhị phân tìm số bí ẩn
- Tạo 1 struct grader với 3 hàm sau:
 - `Init()` → khởi tạo đáp án, gán `ans = 100` hoặc ngẫu nhiên với `rand()`
 - `String Ask(int x)` → nếu `ans < x` thì return `<`, ngược lại in ra `>=`
 - `Yes(int x)` → In ra đáp án! `x`
- Trong hàm `Ask()` và `Yes()` phải viết `assert` để thỏa mãn các điều kiện của đề bài
- `cntAsk <= 25`
- Hàm `yes(int x)` phải kiểm tra `x == ans`

Viết trình chấm cho bài toán Interactive

➤ Code gốc đã AC

```
int lo = 1, hi = 1000;
int ans = -1;
while(lo <= hi) {
    int mid = (lo+hi)/2;
    cout << mid << endl;
    cout << flush;
    string s;
    cin >> s;
    if(s == "<") {
        hi = mid - 1;
    }
    else if (s == ">=") {
        ans = max(ans, mid);
        lo = mid + 1;
    }
}

cout << "! " << ans << endl;
cout << flush;
```

Code với trình chấm

```
string ask(int x) {
    string s;
#ifdef turnOn
    s = gra.ask(x);
#else
    cout << x << endl;
    cout << flush;
    cin >> s;
#endif
    return s;
}

void yes(int x) {
#ifdef turnOn
    gra.yes(x);
#else
    cout << "! " << x << endl;
    cout << flush;
#endif
    exit(0);
}
```

```
#ifdef turnOn
gra.init();
#endif

int lo = 1, hi = 1000;
int ans = -1;
while(lo <= hi) {
    int mid = (lo+hi)/2;
    string s = ask(mid);
    if(s == "<") {
        hi = mid - 1;
    }
    else if (s == ">=") {
        ans = max(ans, mid);
        lo = mid + 1;
    }
}
yes(ans);
```



Viết trình chấm cho bài toán Interactive

➤ Chạy với trình chấm

- init 21
- ANSWER QUERY 500 <
- ANSWER QUERY 250 <
- ANSWER QUERY 125 <
- ANSWER QUERY 62 <
- ANSWER QUERY 31 <
- ANSWER QUERY 15 >=
- ANSWER QUERY 23 <
- ANSWER QUERY 19 >=
- ANSWER QUERY 21 >=
- ANSWER QUERY 22 <
- ! 21
- CORRECT
- cntAsk = 10

Sau đó chạy với chương trình của mình để kiểm tra



Ví dụ 4: Bài K contest miền Trung

- Xét hoán vị của 1, 2, 3, 4, 5, 6.
- Cho N hoán vị ($1 \leq N \leq 719 = 6! - 1$)
- Mỗi truy vấn “? i” cho biết một số của hoán vị thứ N
- In ra cấu hình c1, c2, c3, c4, c5, c6 thỏa mãn
- Số lần gửi yêu cầu không quá 1000.

https://oj.vnoi.info/problem/icpc25_mt_K



Ví dụ 4: Bài K contest miền Trung

- Ý tưởng: xây dần một hoán vị mới $p_1 \dots p_6$.
 - Hỏi cột 1 của tất cả N block \Rightarrow đếm số lần mỗi màu xuất hiện ở vị trí 1. Chọn màu p_1 sao cho đếm $< 5!$ ($=120$). Khi đó chắc chắn còn thiếu một hoán vị bắt đầu bằng p_1 .
 - Chỉ giữ các block có vị trí 1 = p_1 , rồi hỏi vị trí 2 của các block này. Chọn p_2 sao cho đếm $< 4!$ ($=24$).
 - Lặp tương tự cho vị trí 3 (ngưỡng $3!=6$), vị trí 4 ($2!=2$), vị trí 5 ($1!=1$). Màu thứ 6 là màu còn lại.
- Vì mỗi bước ta chọn màu có đếm nhỏ hơn số hoán vị còn lại cho hậu tố, chắc chắn hoán vị tìm được chưa tồn tại trong dãy ban đầu.
- Hỏi cột 1: N lần; cột 2: ≤ 120 ; cột 3: ≤ 24 ; cột 4: ≤ 6 ; cột 5: ≤ 2 .
- Số lượng truy vấn $\leq N + 120 + 24 + 6 + 2 \leq 871$ truy vấn (với $N \leq 719$).