# Huấn luyện viên Tư duy Thuật toán

## Phân tích bài 1328A - Divisibility Problem

Slide Learning CPP

Ngày 20 tháng 1 năm 2026

# Chào bạn!

## Sứ mệnh

Chúng ta sẽ cùng nhau "mổ xẻ"các bài toán theo phong cách **Micro-Chunks**, tập trung hoàn toàn vào chiến thuật và logic thay vì chỉ nhìn vào code.

# Bước 1: Tiếp nhận & Phẫu thuật (Briefing)

- **Đề bài:** Tìm số bước tối thiểu để biến $a$ thành một số chia hết cho $b$.
- **Quy tắc:** Trong mỗi bước, chỉ được phép tăng $a$ lên 1 đơn vị ($a = a + 1$).
- **Mục tiêu:** Tìm số bước tăng ít nhất sao cho $a \pmod b = 0$.

## Lộ trình tư duy

1. **Chunk 1:** Bản chất phép chia hết và "khoảng cách".
2. **Chunk 2:** Xử lý trường hợp đặc biệt.
3. **Chunk 3:** Tối ưu hóa - Tại sao không dùng vòng lặp?

# Chunk 1: Khoảng cách tới "vạch đích"

Hãy tưởng tượng số $b$ giống như **độ dài của một bước chân**. Bạn đang ở vị trí $a$ và muốn nhảy đến một vị trí là bội số của $b$.

## Thử thách tư duy

Nếu $a = 13$ và $b = 4$:

- Các mốc chia hết cho 4 gần đó là: 4, 8, 12, 16, 20...
- Vì bạn chỉ có thể **tăng** $a$, mốc tiếp theo là bao nhiêu?

# Chunk 1: Khoảng cách tới "vạch đích"

Hãy tưởng tượng số $b$ giống như **độ dài của một bước chân**. Bạn đang ở vị trí $a$ và muốn nhảy đến một vị trí là bội số của $b$.

## Thử thách tư duy

Nếu $a = 13$ và $b = 4$:

- Các mốc chia hết cho 4 gần đó là: 4, 8, 12, 16, 20...
- Vì bạn chỉ có thể **tăng** $a$, mốc tiếp theo là bao nhiêu?

**Đáp án:** Mốc tiếp theo là **16**. Cần $16 - 13 = 3$ bước.

# Chunk 2: Công thức "Nhảy cóc"

Trong lập trình thi đấu, nếu $a$ và $b$ rất lớn ($10^9$), dùng vòng lặp `while (a % b != 0) a++;` sẽ bị **TLE** (Time Limit Exceeded).

## Tổng quát hóa

Khi chia $a$ cho $b$:

- Phần nguyên: $a/b$.
- Số dư: $r = a \pmod{b}$.

## Gợi ý

Nếu có 13 cái kẹo ($a = 13$), túi mỗi túi 4 cái ($b = 4$), bạn dư 1 cái ($r = 1$). Cần thêm bao nhiêu cái để đủ 1 túi nữa?

# Chunk 2: Công thức "Nhảy cóc"

Trong lập trình thi đấu, nếu $a$ và $b$ rất lớn ($10^9$), dùng vòng lặp `while (a % b != 0) a++;` sẽ bị **TLE** (Time Limit Exceeded).

## Tổng quát hóa

Khi chia $a$ cho $b$:

- Phần nguyên: $a/b$.
- Số dư: $r = a \pmod{b}$.

## Gợi ý

Nếu có 13 cái kẹo ($a = 13$), túi mỗi túi 4 cái ($b = 4$), bạn dư 1 cái ($r = 1$). Cần thêm bao nhiêu cái để đủ 1 túi nữa?

**Công thức:** $b - (a \pmod{b})$.

# Chunk 3: Xử lý "Bẫy"logic

## Vấn đề

Nếu $a$ đã chia hết cho $b$ ngay từ đầu (VD: $a = 8, b = 4$)?

- Số dư $a \pmod b = 0$.
- Áp dụng công thức $b - (a \pmod b) \Rightarrow 4 - 0 = 4$.
- **Thực tế:** Cần 0 bước.

# Chunk 3: Xử lý "Bẫy"logic

## Vấn đề

Nếu $a$ đã chia hết cho $b$ ngay từ đầu (VD: $a = 8, b = 4$)?

- Số dư $a \pmod b = 0$.
- Áp dụng công thức $b - (a \pmod b) \Rightarrow 4 - 0 = 4$.
- **Thực tế:** Cần 0 bước.

## Giải pháp dùng IF

```
if (a % b == 0) return 0;
else return b - (a % b);
```

# Chunk 4: Công thức "Một dòng"

## Mẹo toán học

Để gộp cả hai trường hợp mà không cần `if`:

```
(b - (a % b)) % b
```

# Chunk 4: Công thức "Một dòng"

## Mẹo toán học

Để gộp cả hai trường hợp mà không cần `if`:

```
(b - (a % b)) % b
```

**Kiểm chứng:**

1. $a = 13, b = 4$: $(4 - (13\%4))\%4 = (4 - 1)\%4 = 3\%4 = 3$.
2. $a = 8, b = 4$: $(4 - (8\%4))\%4 = (4 - 0)\%4 = 4\%4 = 0$.

# Bước cuối: Chốt thuật toán

## Mã giả (Pseudo-code)

```
Nhap t (so luong test case)
Lap t lan:
    Nhap a, b
    Ket qua = (b - (a % b)) % b
    In ra Ket qua
```

## Lưu ý Edge Case

- $a, b \leq 10^9$: Sử dụng kiểu dữ liệu `int` là đủ.
- Luôn chú ý trường hợp $a < b$: Công thức vẫn hoạt động đúng.

**Bạn đã sẵn sàng để tự viết Code chưa?**