

Chương trình này kiểm tra xem một số có phải là số Armstrong không. Dưới đây là giải thích chi tiết từng phần:

## 1. Định nghĩa số Armstrong

Số Armstrong là số mà tổng của các lũy thừa các chữ số của nó (mỗi chữ số nâng lên lũy thừa là số lượng chữ số) bằng chính số đó.


Ví dụ: Số 153 là số Armstrong vì có 3 chữ số, và:

$$1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$$

## 3. Giải thích từng phần của chương trình

### Phần 1: Khai báo thư viện và hàm `isArmstrong` C

c


 Sao chép mã

```
#include <stdio.h>
#include <math.h>
```

- `#include <stdio.h>`: Thư viện này cho phép sử dụng các hàm nhập/xuất cơ bản như `printf` và `scanf`.
- `#include <math.h>`: Thư viện toán học để sử dụng hàm `pow` (lũy thừa).

### Phần 2: Hàm `isArmstrong`

c


 Sao chép mã

```
int isArmstrong(int num) {
    int originalNum, remainder, result = 0, n = 0;
```

- `originalNum`: Lưu trữ bản sao của số đầu vào `num` để tính toán mà không thay đổi giá trị gốc.
- `remainder`: Lưu trữ phần dư (từng chữ số của số `num`).
- `result`: Dùng để tích lũy tổng của các lũy thừa của các chữ số.
- `n`: Biến để đếm số lượng chữ số của `num`.

### Phần 3: Đếm số chữ số

c


 Sao chép mã

```
originalNum = num;
while (originalNum != 0) {
    originalNum /= 10;
    ++n;
}
```

- Đầu tiên, lưu số ban đầu vào `originalNum`.
- Dùng vòng lặp `while` để đếm số chữ số của `num` bằng cách chia liên tục cho 10.
  - Ví dụ, với số 153, ta sẽ chia cho 10 ba lần, vậy  $n = 3$ .

#### Phần 4: Tính tổng các lũy thừa các chữ số

c

 Sao chép mã

```
originalNum = num;
while (originalNum != 0) {
    remainder = originalNum % 10;
    result += pow(remainder, n);
    originalNum /= 10;
}
```


- Gán lại giá trị của `originalNum` là `num` để tiếp tục tính toán.
- Trong vòng lặp này, ta tách từng chữ số của `num` bằng cách lấy phần dư của `originalNum` chia cho 10 (`remainder = originalNum % 10`).
- Sau đó, tính `remainder^n` (chữ số lũy thừa  $n$ ) bằng `pow(remainder, n)` và cộng vào `result`.
- Giảm `originalNum` bằng cách chia cho 10 để chuyển sang chữ số kế tiếp.

Ví dụ với số 153:

- Bước 1: `remainder = 3,  $3^3=27$ , result = 27`
- Bước 2: `remainder = 5,  $5^3=125$ , result = 27 + 125 = 152`
- Bước 3: `remainder = 1,  $1^3=1$ , result = 152 + 1 = 153`

#### Phần 5: Kiểm tra và trả về kết quả

c


 Sao chép mã

```
return result == num;
```

- Nếu `result` (tổng các lũy thừa của các chữ số) bằng `num`, thì hàm trả về 1 (true), nghĩa là `num` là số Armstrong. Nếu không, trả về 0 (false).

## Phần 6: Hàm `main`

c

 Sao chép mã

```
int main() {
    int num;
    printf("Nhập một số: ");
    scanf("%d", &num);

    if (isArmstrong(num)) {
        printf("%d là số Armstrong.\n", num);
    } else {
        printf("%d không phải là số Armstrong.\n", num);
    }
    return 0;
}
```

- Nhận đầu vào `num` từ người dùng.
- Gọi hàm `isArmstrong(num)` để kiểm tra. Nếu kết quả là `true`, in ra rằng `num` là số Armstrong, nếu không thì ngược lại.