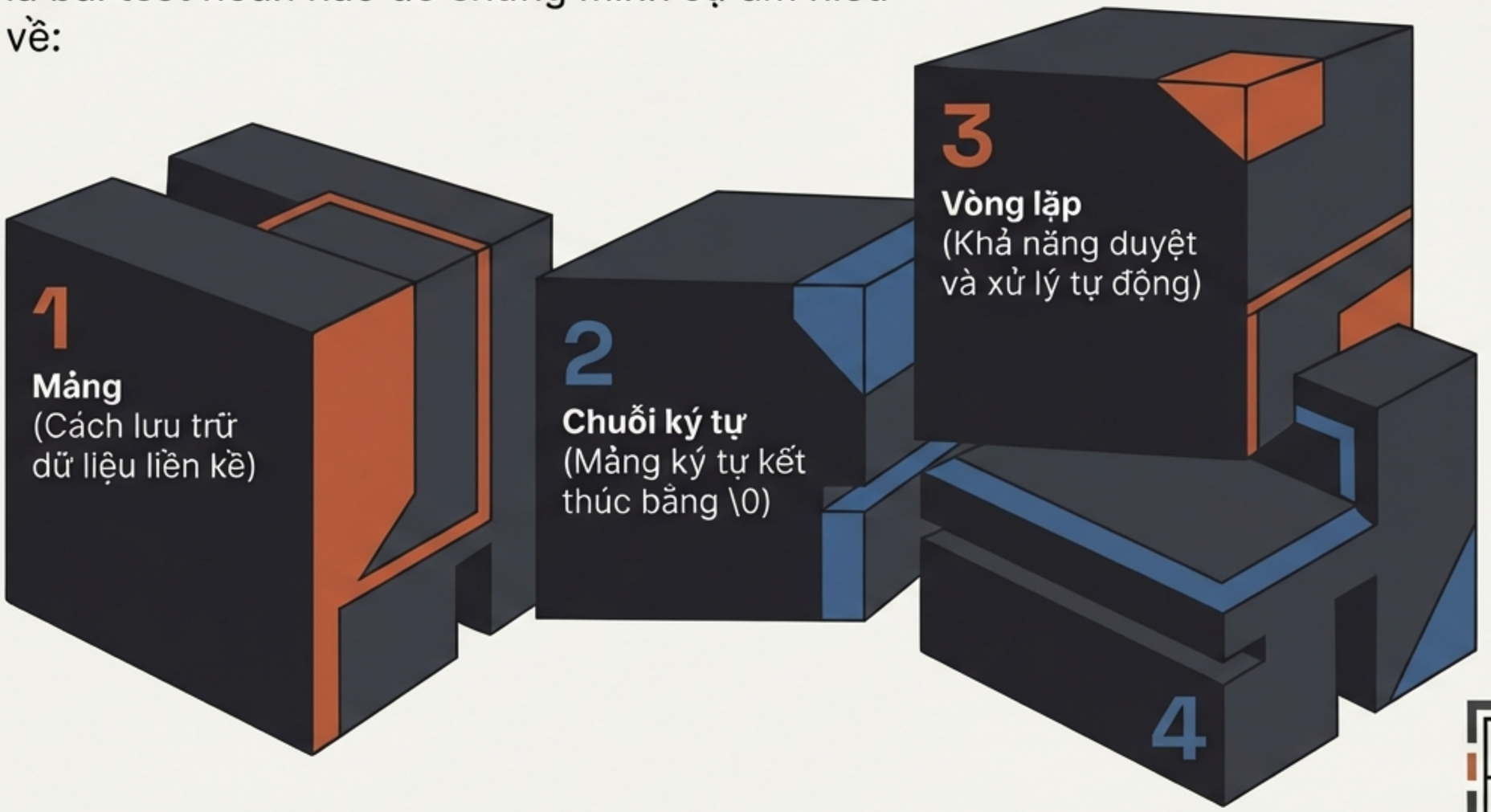
Giải mã dữ liệu và phân loại ký tự trong C

	0x41	0x41	48	48	1
0x5A	0x5A	0x5A	48	65	0
0x51	0x51	0x51	48	65	0
0x4D	0x4D	0x4D	65	97	1
0x4B	0x4B	0x4B	65	97	0
0x48	0x48	0x48	97	100	1
0x52	0x52	0x52	97	100	0
0x50	0x50	0x50	48	100	7F
	0x58	0x58	65	100	7F

Error404-Labs.Info.Vn

Bài toán nền tảng thử thách 4 kỹ năng cốt lõi

Đếm số lần xuất hiện của mỗi ký tự trong chuỗi là bài test hoàn hảo để chứng minh sự am hiểu về:



Error404-Labs.Info.Vn

Xác định rõ đầu vào và đầu ra mong muốn

Người dùng gõ từ bàn phím
một chuỗi bất kỳ.

```
Input: hello
```

Hệ thống đếm và in ra màn hình
tần số của từng ký tự riêng biệt.

```
h - 1
e - 1
l - 2
o - 1
```

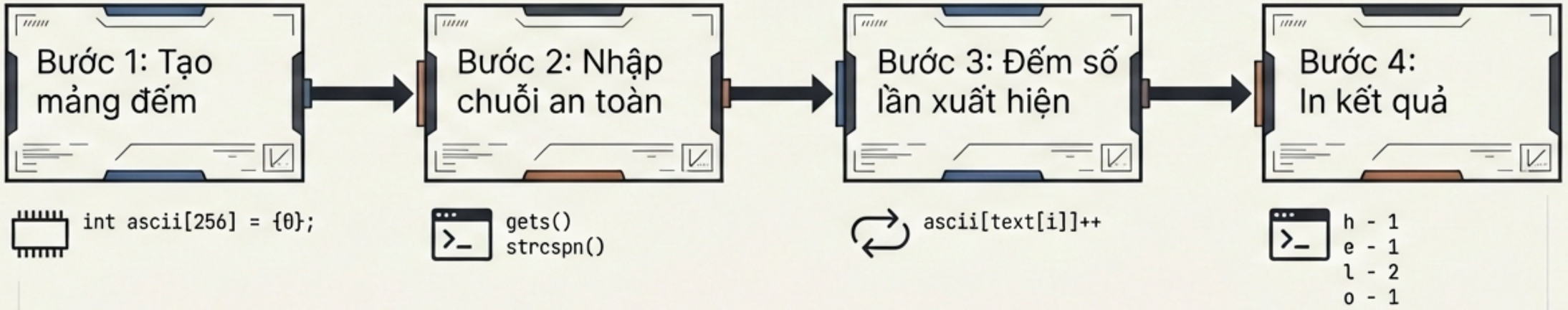
Error404-Labs.Info.Vn

Bản thiết kế tổng thể của thuật toán phân loại

```c
#include <stdio.h>
#include <string.h>

int main() {
    int ascii[256] = {0};
    char text[1000];

    printf("\nInput: ");
    fgets(text, sizeof(text), stdin);
    text[strcspn(text, "\n")] = '\0';

    printf("\nOutput: %s", text);
    for(int i = 0; text[i] != '\0'; i++) {
        ascii[text[i]]++;
    }

    printf("\nResult: ");
    for(int i = 0; i < 256; i++) {
        if(ascii[i] > 0) {
            printf("\n%c - %d", i, ascii[i]);
        }
    }
    return 0;
}
```

Error404-Labs.Info.Vn

Bóc tách thuật toán qua 4 giai đoạn xử lý

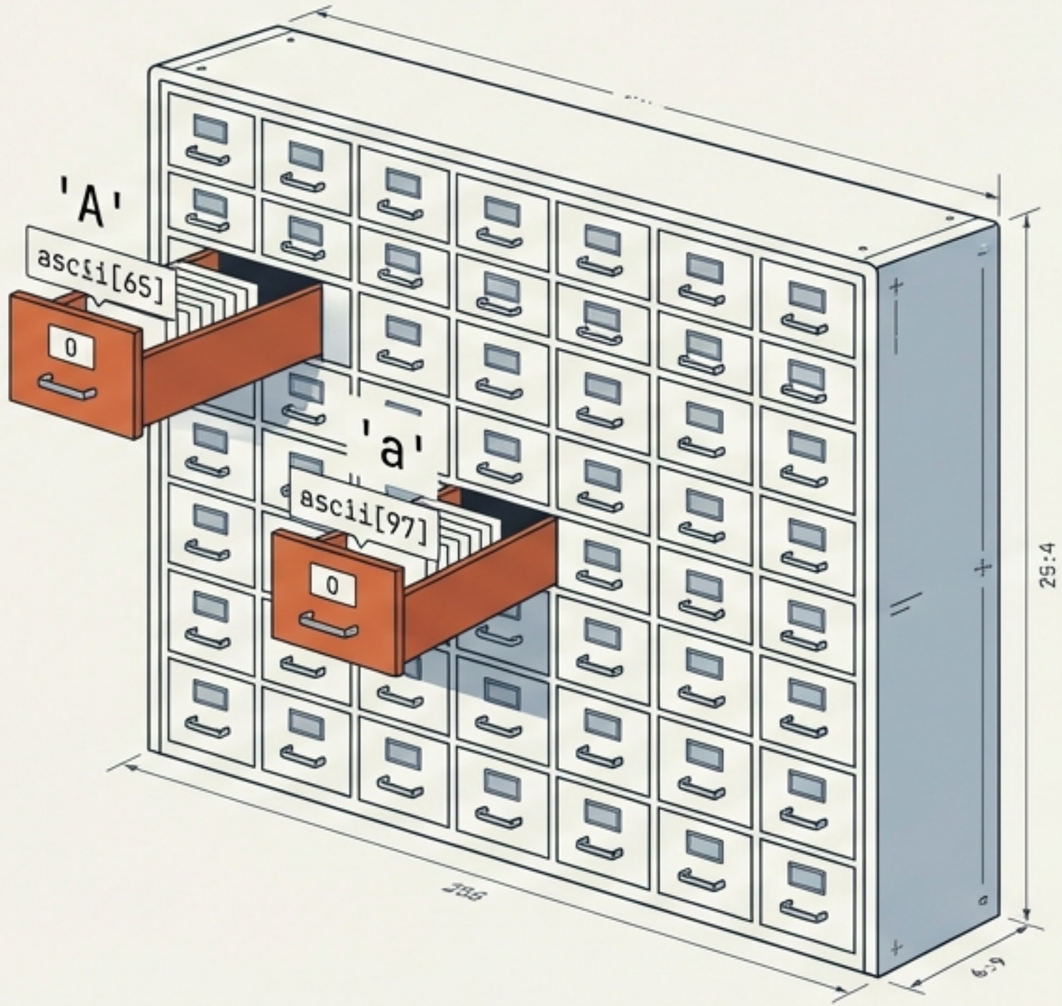


Error404-Labs.Info.Vn

Giai đoạn 1 - Khởi tạo tủ tài liệu 256 ngăn

```
int ascii[256] = {0};
```

- Bảng mã ASCII có đúng 256 ký tự (giá trị từ 0 đến 255).
- Ta tạo một mảng 256 phần tử, đóng vai trò như 256 “ngăn kéo” lưu số lần xuất hiện.
- Khởi tạo toàn bộ ngăn kéo với giá trị ban đầu là 0.

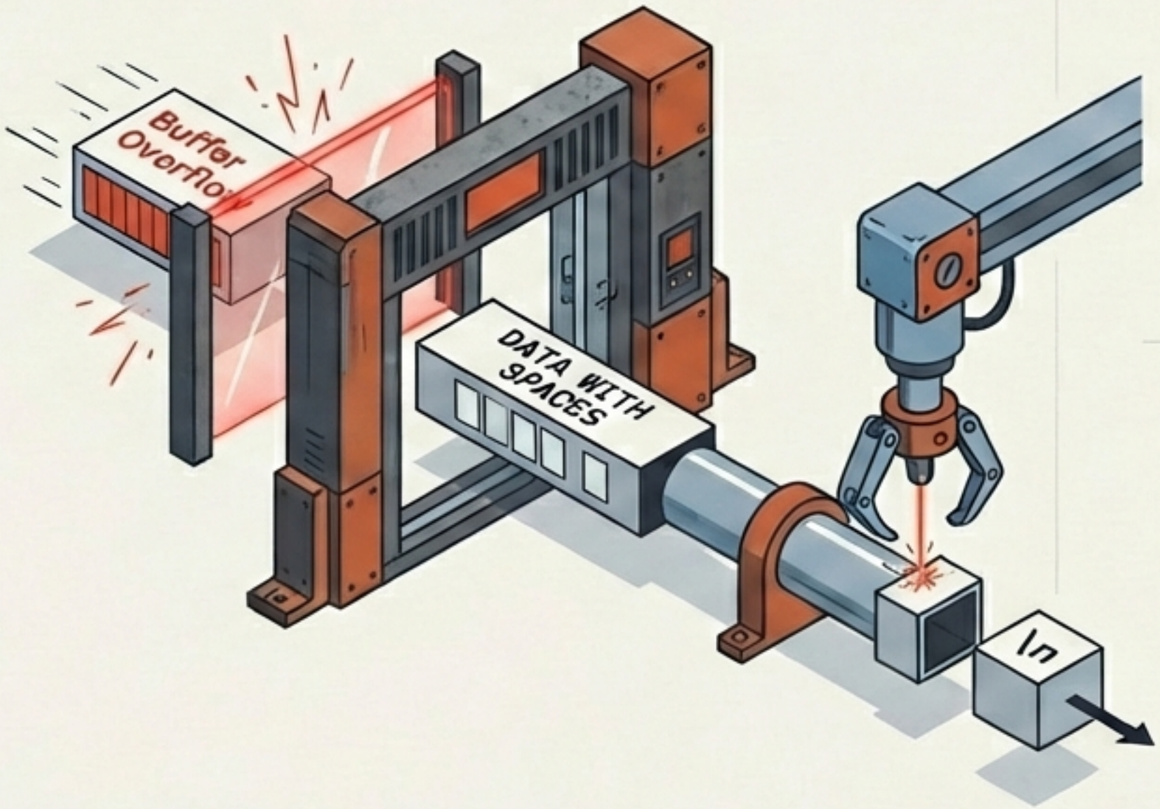


Giai đoạn 2 - Thiết lập cánh cửa an ninh kiểm soát đầu vào

```
fgets(text, sizeof(text), stdin);
text[strcspn(text, "\n")] = '\0';
```

Sự vượt trội của fgets so với scanf("%s"):
- Đọc được cả khoảng trắng (dấu cách).
- Khống chế kích thước, tránh lỗi tràn bộ nhớ (Buffer Overflow).
- Mức độ bảo mật và an toàn cao hơn.

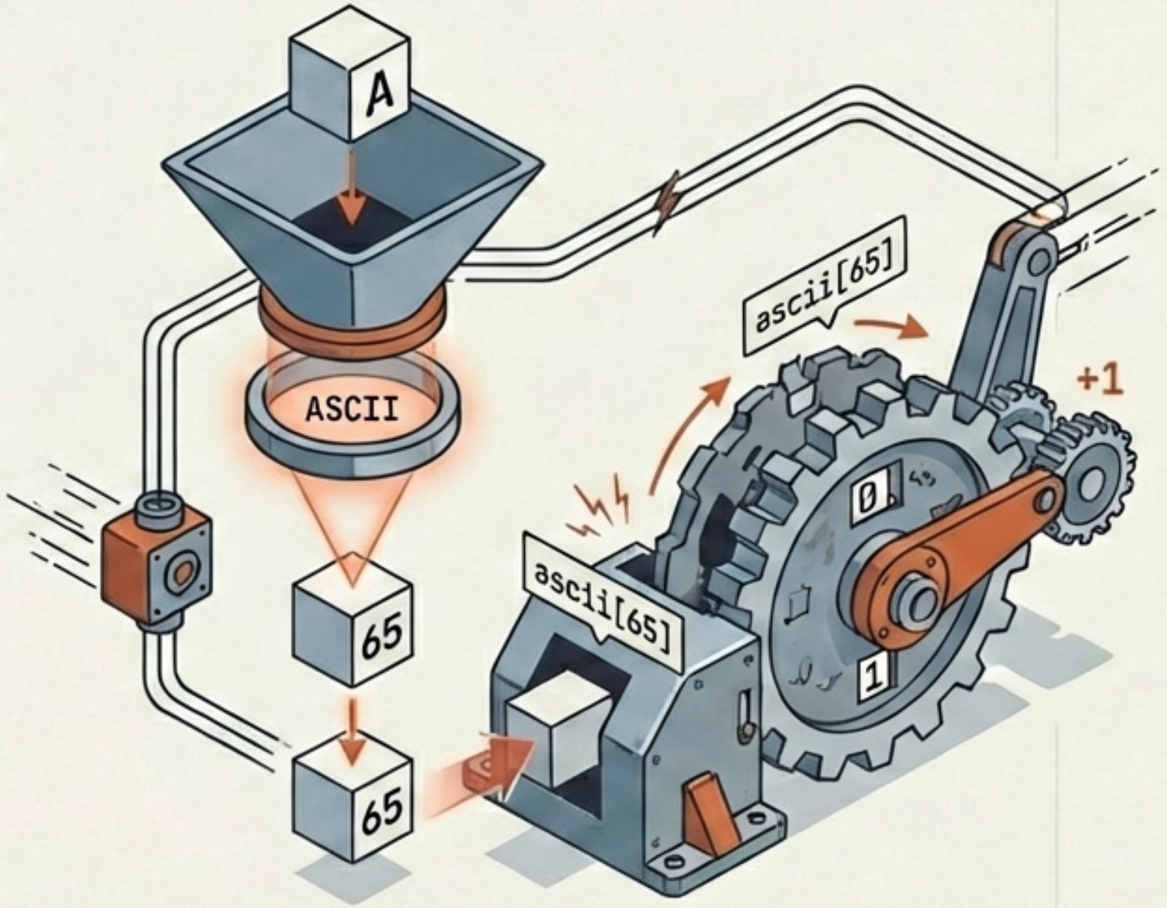
Xử lý mảng: Loại bỏ ký tự xuống dòng \n dư thừa để dữ liệu chuỗi hoàn toàn sạch trước khi xử lý.



Giai đoạn 3 - Vận hành động cơ phân loại dữ liệu

```
for(int i = 0; text[i] != '\0'; i++) {
    ascii[text[i]]++;
}
```

Cơ chế hoạt động:
- Duyệt tuần tự từng ký tự trong chuỗi cho đến khi gặp điểm dừng \0.
- Ép kiểu ngầm định: Lấy mã ASCII của ký tự đang xét để làm vị trí chỉ mục (index).
- Tăng giá trị tại vị trí ngăn kéo tương ứng trong mảng lên 1 đơn vị.



Trực quan hóa quá trình đếm tần số của chuỗi "hello"

Ký tự h → ASCII 104 → Thực thi: ascii[104]++

Ký tự e → ASCII 101 → Thực thi: ascii[101]++

Ký tự l → ASCII 108 → Thực thi: ascii[108]++

Ký tự l → ASCII 108 → Thực thi: ascii[108]++

Ký tự o → ASCII 111 → Thực thi: ascii[111]++

Giai đoạn 4 - Quét và trích xuất kết quả cuối cùng

```
for(int i = 0; i < 256; i++) {
    if(ascii[i] > 0) {
        printf("\n%c - %d", i, ascii[i]);
    }
}
```

Cơ chế trích xuất:
- Quét lại toàn bộ 256 ngăn kéo của tủ tài liệu từ đầu đến cuối.
- Sử dụng điều kiện lọc if(ascii[i] > 0): Chỉ mở và in ra màn hình những ngăn kéo có chứa dữ liệu (tần số xuất hiện lớn hơn 0).

Đánh Giá Độ Phức Tạp Và Hiệu Năng Của Thuật Toán

O(n)

Thời gian (Time Complexity)

Chỉ duyệt qua chuỗi một lần duy nhất với n là chiều dài của chuỗi đầu vào. Rất nhanh và tối ưu.

O(1)

Bộ nhớ (Space Complexity)

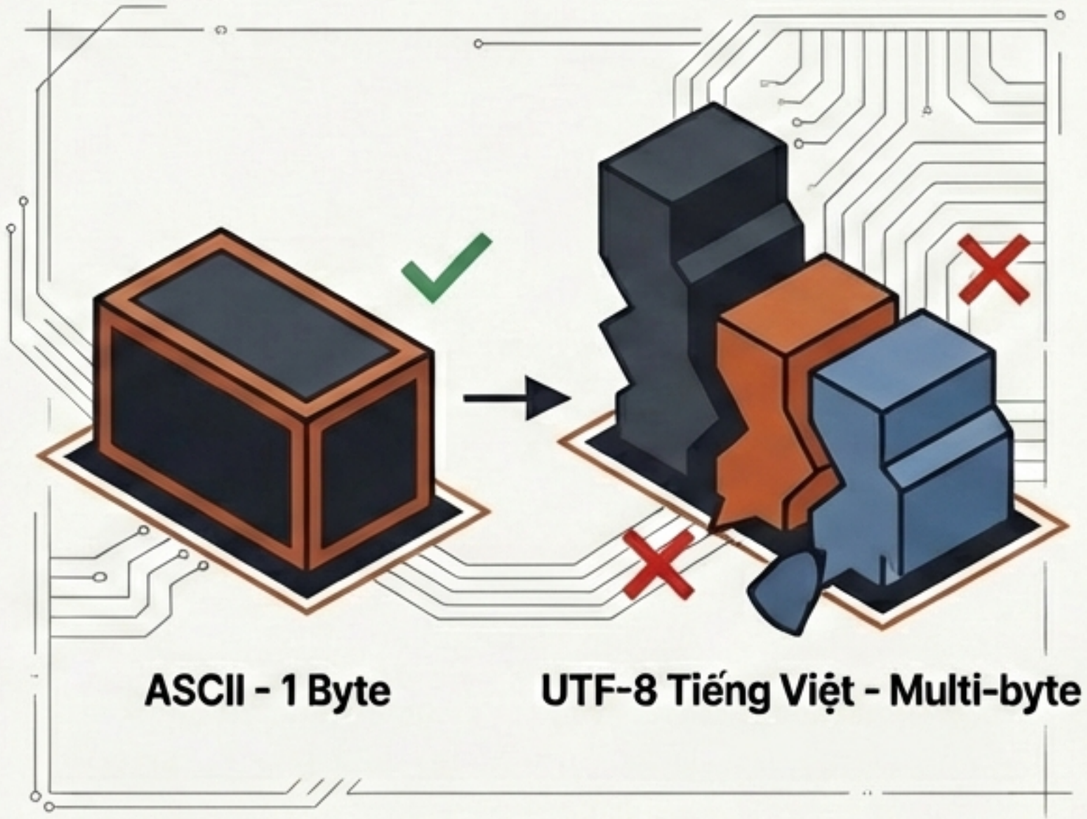
Luôn sử dụng một mảng cố định 256 phần tử, không phụ thuộc vào độ dài của chuỗi nhập vào dài hay ngắn.

Kết luận kỹ thuật: Đây là cách tiếp cận cực kỳ tối ưu cho bài toán đếm tần số cơ bản.

Giới Hạn Thực Tế Với Hệ Thống Mã Hóa UTF-8

Giới hạn hệ thống:
- Thuật toán này hoạt động hoàn hảo với dải ký tự ASCII truyền thống (Tiếng Anh, số, ký tự đặc biệt cơ bản).

Lỗ hổng đa ngôn ngữ:
- Nếu nhập chuỗi Tiếng Việt có dấu (sử dụng bảng mã UTF-8), kết quả sẽ bị sai lệch.
- Nguyên nhân: Chương trình đang xử lý cấu trúc theo từng byte độc lập chứ không nhận diện theo cụm ký tự (multi-byte character).



Làm chủ nghệ thuật xử lý chuỗi và bộ nhớ trong C

- **Cấu trúc dữ liệu:** Dùng mảng cố định để ánh xạ trực tiếp (Direct Mapping).
- **Thuật toán:** Tận dụng mã ASCII làm chỉ mục để đạt giới hạn tốc độ cao nhất O(1).
- **Bảo mật:** Luôn kiểm soát đầu vào an toàn bằng bộ lọc bộ nhớ.