**Đoạn code lỗi :**

#include <stdio.h>

int main() {

int arr[3] = {1, 2, 3};

printf("%d\n", arr[3]); // Lỗi truy cập ngoài mảng

return 0;

}

**Thông báo lỗi (runtime error / stack trace minh họa)**

Trên nhiều hệ thống (Linux, dùng AddressSanitizer hoặc gdb) có thể thấy:

==12345==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7ffdc1b4a08c at pc 0x7f8b8b5bdf3a bp 0x7ffdc1b4a030 sp 0x7ffdc1b4a028

READ of size 4 at 0x7ffdc1b4a08c thread T0

#0 0x7f8b8b5bdf39 in printf (/usr/lib/libc.so.6+0x5df39)

#1 0x55a2e0b1111a in main (/a.out+0x1111a)

#2 0x7f8b8b59b24a in \_\_libc\_start\_main (/usr/lib/libc.so.6+0x2324a)

#3 0x55a2e0b1100e in \_start (/a.out+0x1100e)

Address 0x7ffdc1b4a08c is located in stack of thread T0 at offset 12 in frame

#0 0x55a2e0b110b7 in main (/a.out+0x110b7)

**3. Prompt gửi cho AI**

“Hãy phân tích nguyên nhân lỗi từ stack trace ở trên, giải thích bằng ngôn ngữ đơn giản dễ hiểu (như cho người mới học lập trình C), sau đó đưa ra bản code đã sửa và giải thích tại sao sửa như vậy.”

**4. Phản hồi mẫu của AI (theo đúng prompt trên)**

**Phân tích nguyên nhân lỗi**

* Mảng arr được khai báo có **3 phần tử**: arr[0], arr[1], arr[2].
* Dòng printf("%d\n", arr[3]); cố **truy cập phần tử thứ 4**, nhưng **mảng chỉ có 3 phần tử hợp lệ (0–2)**.
* Truy cập arr[3] nghĩa là chương trình đang đọc **vùng nhớ nằm ngoài phạm vi được cấp cho mảng**, nên trình biên dịch không ngăn được, nhưng khi chạy (runtime) hệ điều hành phát hiện hành vi truy cập “ngoài vùng được phép” → **stack-buffer-overflow** → chương trình có thể **crash hoặc in ra giá trị rác**.

**Sơ đồ minh họa bộ nhớ ngăn xếp (stack)**

Stack memory:

+------------+

| arr[0] = 1 | <- index 0

+------------+

| arr[1] = 2 | <- index 1

+------------+

| arr[2] = 3 | <- index 2

+------------+

| arr[3] = ? | <- ngoài phạm vi, không thuộc arr

+------------+

=> arr[3] truy cập ra ngoài vùng hợp lệ -> lỗi.

**5. Bản code đã sửa**

**Cách 1 – Truy cập đúng phạm vi**

#include <stdio.h>

int main() {

int arr[3] = {1, 2, 3};

printf("%d\n", arr[2]); // Truy cập phần tử cuối hợp lệ

return 0;

}

**Cách 2 – Duyệt mảng bằng vòng lặp an toàn**

#include <stdio.h>

int main() {

int arr[3] = {1, 2, 3};

for (int i = 0; i < 3; i++) {

printf("%d\n", arr[i]); // chỉ in các phần tử hợp lệ

}

return 0;

}

**Giải thích sửa lỗi**

* Mỗi mảng arr có chỉ số hợp lệ từ 0 → size - 1.
* Khi dùng chỉ số bằng hoặc lớn hơn kích thước mảng (3 trong ví dụ này), ta truy cập vùng nhớ không thuộc mảng → **undefined behavior**.
* Sửa bằng cách **đảm bảo chỉ số nằm trong phạm vi hợp lệ**, hoặc **lặp từ 0 đến < kích thước mảng**.