## ĐỀ BÀI

Sinh viên chọn một hàm đơn giản (ví dụ: kiểm tra số nguyên tố, tính giai thừa, tính tổng mảng). Dạng dấu nhắc mẫu:  
“Viết hàm C kiểm tra số nguyên tố, có chú thích từng bước, đầu vào là số nguyên dương n.”  
  
Thử thay đổi cách diễn đạt (dài hơn/ngắn hơn) để so sánh kết quả AI trả về. Có ít nhất 2 lời nhắc (prompt) phiên bản được đưa ra. So sánh kết quả mã sinh của AI và chỉ ra các điểm khác biệt.

## PHẦN 1 – CÁC LỜI NHẮC (PROMPT)

### Prompt 1 (ngắn gọn)

Viết hàm C kiểm tra số nguyên tố, có chú thích từng bước, đầu vào là số nguyên dương n.

### Prompt 2 (mô tả dài hơn)

Hãy viết một chương trình C hoàn chỉnh gồm hàm kiem\_tra\_nguyen\_to(int n) có nhiệm vụ xác định xem n có phải là số nguyên tố hay không. Trong hàm, giải thích bằng chú thích từng bước (bằng tiếng Việt), từ việc nhập dữ liệu, kiểm tra điều kiện biên, đến khi kết luận kết quả. In ra thông báo “n là số nguyên tố” hoặc “n không phải là số nguyên tố”.

## PHẦN 2 – KẾT QUẢ AI TRẢ VỀ

### Kết quả cho Prompt 1

#include <stdio.h>  
#include <stdbool.h>  
  
// Hàm kiểm tra số nguyên tố  
bool isPrime(int n) {  
 if (n < 2) return false; // Số nhỏ hơn 2 không phải là số nguyên tố  
 for (int i = 2; i \* i <= n; i++) { // Duyệt các ước từ 2 đến căn bậc hai của n  
 if (n % i == 0) return false; // Nếu chia hết thì không phải số nguyên tố  
 }  
 return true; // Nếu không chia hết, là số nguyên tố  
}  
  
int main() {  
 int n;  
 scanf("%d", &n);  
 if (isPrime(n))  
 printf("%d là số nguyên tố\n", n);  
 else  
 printf("%d không phải là số nguyên tố\n", n);  
 return 0;  
}

### Kết quả cho Prompt 2

#include <stdio.h>  
#include <stdbool.h>  
  
// Hàm kiểm tra số nguyên tố  
bool kiem\_tra\_nguyen\_to(int n) {  
 // Bước 1: Kiểm tra đầu vào  
 if (n < 2) {  
 // Các số nhỏ hơn 2 không phải là số nguyên tố  
 return false;  
 }  
  
 // Bước 2: Kiểm tra ước từ 2 đến căn bậc hai của n  
 for (int i = 2; i \* i <= n; i++) {  
 // Nếu n chia hết cho i thì n không phải số nguyên tố  
 if (n % i == 0) {  
 return false;  
 }  
 }  
  
 // Bước 3: Nếu không tìm thấy ước nào, n là số nguyên tố  
 return true;  
}  
  
int main() {  
 int n;  
 printf("Nhập số nguyên dương n: ");  
 scanf("%d", &n);  
  
 // Gọi hàm và in kết quả  
 if (kiem\_tra\_nguyen\_to(n))  
 printf("%d là số nguyên tố.\n", n);  
 else  
 printf("%d không phải là số nguyên tố.\n", n);  
  
 return 0;  
}

## PHẦN 3 – SO SÁNH KẾT QUẢ AI TRẢ VỀ

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | Prompt 1 (ngắn) | Prompt 2 (dài) | Nhận xét |
| Cấu trúc chương trình | Chỉ gồm hàm isPrime() và main() tối giản | Có cấu trúc đầy đủ, rõ từng bước | Prompt dài giúp AI sinh code rõ ràng hơn |
| Ngôn ngữ chú thích | Ngắn gọn, chỉ chú thích dòng chính | Có chú thích chi tiết theo từng “bước” | Prompt dài thực hiện đúng yêu cầu “từng bước” |
| Cách đặt tên biến/hàm | Dùng tiếng Anh (isPrime) | Dùng tiếng Việt (kiem\_tra\_nguyen\_to) | AI điều chỉnh theo ngôn ngữ mô tả |
| Mức độ tương tác với người dùng | Không có hướng dẫn nhập | Có lời nhắc nhập dữ liệu | Prompt dài → AI thêm hướng dẫn nhập/xuất |
| Phong cách lập trình | Cơ bản, ngắn gọn | Tường minh, mang tính sư phạm | Phù hợp khi dạy học hoặc trình bày báo cáo |

## KẾT LUẬN

- Prompt ngắn cho kết quả nhanh, code tối giản, nhưng phần giải thích và chú thích ít.  
- Prompt dài, chi tiết khiến AI sinh ra mã dễ hiểu, có cấu trúc rõ ràng và gần với bài mẫu cho sinh viên.  
- Khi dùng AI để học lập trình, nên mô tả rõ yêu cầu, phong cách trình bày, ngôn ngữ chú thích để nhận được kết quả tối ưu.