## Prompt 1

Viết hàm C tính tổng các số lẻ nhỏ hơn n.

#include <stdio.h>  
  
int tongLe(int n){  
 int s=0;  
 for(int i=1;i<n;i++){  
 if(i%2!=0)  
 s+=i;  
 }  
 return s;  
}  
  
int main(){  
 int n;  
 printf("Nhap n: ");  
 scanf("%d",&n);  
 printf("Tong cac so le nho hon %d la: %d\n",n,tongLe(n));  
 return 0;  
}

## Prompt 2

Viết hàm C tính tổng các số lẻ nhỏ hơn n. Code phải dễ đọc, đặt tên biến rõ ràng và có chú thích từng dòng lệnh.

#include <stdio.h>  
  
// Hàm tính tổng các số lẻ nhỏ hơn n  
int tinhTongSoLe(int n) {  
 int tong = 0; // Khởi tạo biến tổng  
 for (int i = 1; i < n; i++) { // Duyệt từ 1 đến n-1  
 if (i % 2 != 0) { // Kiểm tra nếu i là số lẻ  
 tong += i; // Cộng i vào tổng  
 }  
 }  
 return tong; // Trả về kết quả tổng  
}  
  
int main() {  
 int n;  
 printf("Nhập số nguyên dương n: "); // Nhập dữ liệu từ người dùng  
 scanf("%d", &n);  
  
 int ketQua = tinhTongSoLe(n); // Gọi hàm và lưu kết quả  
 printf("Tổng các số lẻ nhỏ hơn %d là: %d\n", n, ketQua); // In kết quả  
 return 0;  
}

So sánh sự khác biệt giữa hai kết quả

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | Prompt 1 (Cơ bản) | Prompt 2 (Chi tiết, dễ đọc) | Nhận xét |
| Tên hàm & biến | tongLe, s, i | tinhTongSoLe, tong, ketQua, i | Phiên bản 2 đặt tên rõ nghĩa, dễ hiểu hơn |
| Chú thích | Không có | Có chú thích từng dòng | Giúp người đọc hiểu từng bước xử lý |
| Cấu trúc code | Gọn nhưng khó đọc | Căn chỉnh đẹp, logic rõ | Phiên bản 2 tuân thủ coding style tốt |
| Mức độ phù hợp với người mới học | Trung bình | Rất dễ học theo | Phiên bản 2 hỗ trợ học sinh dễ hiểu |
| Độ dài code | Ngắn hơn | Dài hơn nhưng có chú thích | Thêm chú thích làm tăng độ dài nhưng cải thiện chất lượng học tập |

## Kết luận

Việc thêm yêu cầu 'code dễ đọc, đặt tên biến rõ ràng' và 'có chú thích từng dòng' khiến AI sinh ra code có phong cách viết tốt hơn, tên biến có ý nghĩa và bố cục rõ ràng. Mặc dù dài hơn, phiên bản này hữu ích hơn trong môi trường học tập hoặc khi đọc lại code.