## Code gốc

#include <stdio.h>  
int main(){int n,i,j,s=0,f;scanf("%d",&n);for(i=2;i<=n;i++){f=0;for(j=2;j<i;j++){if(i%j==0){f=1;break;}}if(!f)s+=i;}printf("%d",s);return 0;}

## Code refactor (có chú thích)

#include <stdio.h>  
  
// Function to check if a number is prime  
int isPrime(int num) {  
 if (num < 2) return 0;  
 for (int i = 2; i \* i <= num; i++) {  
 if (num % i == 0) return 0;  
 }  
 return 1;  
}  
  
int main() {  
 int n, sum = 0;  
 scanf("%d", &n);  
  
 // Calculate the sum of prime numbers up to n  
 for (int i = 2; i <= n; i++) {  
 if (isPrime(i)) {  
 sum += i;  
 }  
 }  
  
 printf("%d", sum);  
 return 0;  
}

## Phân tích & Điểm cải thiện

\*\*So sánh & Điểm cải thiện:\*\*  
  
1. \*\*Đặt tên biến rõ ràng hơn:\*\*   
 - `s` → `sum`, `f` → bỏ, dùng hàm `isPrime` để tách chức năng.  
 - Giúp dễ hiểu mục đích của từng biến.  
  
2. \*\*Tách hàm kiểm tra số nguyên tố:\*\*   
 - Từ vòng lặp lồng nhau trong `main` → thành hàm riêng `isPrime()`.  
 - Tăng khả năng tái sử dụng và dễ kiểm thử.  
  
3. \*\*Cấu trúc chương trình rõ ràng:\*\*   
 - Thêm chú thích từng phần, thụt lề đúng chuẩn.  
 - Code dễ đọc, dễ bảo trì hơn.  
  
4. \*\*Hiệu năng tốt hơn:\*\*   
 - Kiểm tra ước số tới `sqrt(num)` thay vì tới `num - 1`.  
 - Giảm thời gian chạy khi n lớn.  
  
\*\*Kiểm chứng logic:\*\*   
Cả hai đoạn code đều cho cùng kết quả với cùng giá trị đầu vào `n`.   
Bản refactor chạy nhanh hơn, dễ đọc hơn, dễ bảo trì hơn.