

## **Yêu cầu 1:**

Bạn là sinh viên CNTT và cần học một framework mới (ví dụ: Django hoặc ReactJS). Hãy trình bày cách bạn dùng AI để xây dựng lộ trình học tập chi tiết, bao gồm: tài liệu, bài tập, và cách đánh giá tiến bộ

## **Prompt:**

Bạn là một giảng viên công nghệ thông tin. Hãy xây dựng lộ trình học ReactJS chi tiết dành cho sinh viên CNTT năm nhất.

Trình bày theo tuần hoặc giai đoạn (có số tuần rõ ràng, ví dụ: Tuần 1–2, Tuần 3–4...).

Nói sơ qua về kiến thức nền tảng (HTML, CSS, JavaScript) trước khi vào ReactJS.

Có dự án nhỏ theo từng giai đoạn và 1 dự án cuối khóa.

Nội dung mỗi tuần/giai đoạn phải có:

- Kiến thức lý thuyết cần nắm (ngắn gọn, liệt kê).
- Tài liệu tham khảo (docs chính thức, video, sách, blog uy tín).
- Bài tập thực hành (có ví dụ từ dễ đến khó).
- Cách đánh giá tiến bộ (checklist kỹ năng và mini test hoặc project).

Nguyên tắc giảng dạy

Trình bày ngắn gọn, rõ ràng, dễ

Ưu tiên học qua thực hành: ít lý thuyết dài dòng, nhiều coding

Hướng dẫn sinh viên tự học thông qua tài liệu chính thống + project thực tế

Có lời khuyên hoặc mẹo học tập ở từng giai đoạn

Tổng kết cuối khóa

Yêu cầu sinh viên có thể xây dựng 1 ứng dụng ReactJS nhỏ hoàn chỉnh (ví dụ: Todo App, Blog, E-commerce mini).

Đưa ra tiêu chí đánh giá cuối cùng: kiến thức, kỹ năng, sản phẩm

Hãy trình bày kết quả một cách khoa học, sạch đẹp, dễ áp dụng ngay trong lớp học

## **Yêu cầu 2:**

Viết prompt để AI đóng vai người hướng dẫn học tiếng Anh giao tiếp, với bối cảnh: bạn đi du lịch ở sân bay quốc tế. AI cần tạo đoạn hội thoại mẫu + đưa ra feedback khi bạn trả lời sai.

## Prompt:

Bạn là giáo viên tiếng Anh .Lấy bối cảnh tôi đang đi du lịch ở sân bay quốc tế.

Nhiệm vụ của bạn:

- Đóng vai nhân viên sân bay (check-in, hải quan, an ninh, boarding gate...) và tạo đoạn hội thoại mẫu ngắn gọn (2–6 câu).
- Sau khi đưa ví dụ, đặt câu hỏi để tôi trả lời lại bằng tiếng Anh.
- Nếu tôi trả lời chưa đúng, thiếu tự nhiên, hoặc sai ngữ pháp, hãy đưa ra feedback chi tiết:
  - o Chỉ ra lỗi (từ vựng, phát âm, ngữ pháp, cách diễn đạt).
  - o Đưa ra phiên bản câu trả lời chuẩn.
  - o Giải thích ngắn gọn vì sao câu trả lời đó đúng.
- Nếu tôi trả lời đúng, hãy đưa ra lời khen và gợi ý cách diễn đạt khác tự nhiên hơn.
- Giữ văn phong thân thiện, dễ hiểu, khích lệ, như một người thầy hướng dẫn trong thực tế

## Yêu cầu 3:

Viết một prompt yêu cầu AI **giải thích khái niệm kiến trúc Client–Server** ở mức độ dễ hiểu cho học sinh cấp 3.

## Prompt:

Hãy giải thích khái niệm kiến trúc Client–Server một cách đơn giản, dễ hiểu như đang dạy cho học sinh cấp 3. Tránh dùng nhiều thuật ngữ phức tạp, hãy kể câu chuyện hoặc đưa ra ví dụ đời sống thực tế để học sinh dễ hình dung, chẳng hạn như việc gọi món trong quán ăn, mượn sách trong thư viện hoặc nhắn tin qua điện thoại. Trình bày thành một đoạn văn mạch lạc, không dùng gạch đầu dòng. Giải thích rõ Client là gì, Server là

gì, cách chúng tương tác với nhau và tại sao mô hình này lại quan trọng trong công nghệ ngày nay. Hãy cố gắng khiến lời giải thích mang tính gần gũi, dễ nhớ và tạo hứng thú cho học sinh

#### **Yêu cầu 4:**

Viết một prompt yêu cầu AI sinh ví dụ code **hàm kiểm tra số nguyên tố**, đồng thời yêu cầu AI giải thích cùng khái niệm này ở **2 góc độ khác nhau**: (1) cho người không học CNTT, (2) cho sinh viên năm cuối. **[10 điểm]**

#### **Prompt:**

Bạn là giảng viên CNTT. Nhiệm vụ của bạn là viết một đoạn code C++ nhận vào hai số nguyên dương a và b, trả về tổng của chúng. Sau đó hãy giải thích khái niệm “tính tổng hai số” ở hai mức độ khác nhau: với người không học CNTT thì giải thích đơn giản, gần gũi với đời sống; với sinh viên CNTT năm cuối thì giải thích theo cách chuyên sâu, có liên hệ tới kiểu dữ liệu số nguyên, cách máy tính thực hiện phép cộng và phân tích độ phức tạp.

Bài học dùng cho cả người mới bắt đầu và sinh viên nâng cao. Luôn trình bày code trước, giải thích sau. Code phải ngắn gọn, không dùng thư viện ngoài. Giải thích cho người không học CNTT phải tránh hoàn toàn thuật ngữ kỹ thuật, còn giải thích cho sinh viên năm cuối phải có ngôn ngữ chuyên ngành và phân tích chính xác.

Đầu ra cần có đoạn code trong khối C++ Hai đoạn văn bản mạch lạc với tiêu đề rõ ràng “Giải thích cho người không học CNTT” và “Giải thích cho sinh viên CNTT năm cuối”.

Input là hai số nguyên dương a và b. Output là code Python cộng hai số cùng với hai đoạn giải thích theo đúng yêu cầu.

#### **Yêu cầu 5:**

Tình huống: Bạn cần chuẩn bị bài thuyết trình 10 phút về chủ đề “Lợi ích và rủi ro khi sử dụng AI trong học tập”. Hãy viết prompt chi tiết để AI:

Tạo outline 3 phần (Mở đầu – Nội dung – Kết luận)

Gợi ý 5 slide với nội dung gạch đầu dòng

Thêm ví dụ minh họa

Viết script ngắn (~2 phút) để luyện nói phần mở đầu

## Prompt:

Bạn là một chuyên gia giáo dục và diễn giả, nhiệm vụ của bạn là hỗ trợ tôi chuẩn bị bài thuyết trình 10 phút với chủ đề “Lợi ích và rủi ro khi sử dụng AI trong học tập”. Mục tiêu là giúp tôi có một bài thuyết trình rõ ràng, dễ hiểu, có ví dụ minh họa và phần mở đầu hấp dẫn để luyện tập nói. Ngữ cảnh là tôi sẽ trình bày trước lớp học đại học với sinh viên CNTT, vì vậy nội dung cần súc tích, mang tính học thuật nhưng không quá hàn lâm để người nghe dễ tiếp thu. Bạn cần tạo một format ba phần gồm Mở đầu, Nội dung, Kết luận; sau đó gợi ý năm slide thuyết trình kèm nội dung ngắn gọn dưới dạng gạch đầu dòng; bổ sung thêm ít nhất một ví dụ minh họa cụ thể để làm rõ vấn đề; cuối cùng viết một đoạn script dài khoảng hai phút để tôi dùng luyện nói cho phần Mở đầu.

Đầu ra phải được định dạng mạch lạc, dễ đọc. Outline trình bày theo ba phần rõ ràng. Nội dung gợi ý cho slide phải ngắn gọn, ưu tiên ý chính thay vì câu dài. Ví dụ minh họa cần thực tế và liên quan trực tiếp đến chủ đề học tập. Script phần Mở đầu nên ở dạng văn bản hoàn chỉnh, có ngôn từ gần gũi, giúp tôi dễ luyện tập khi nói trước đám đông.

Toàn bộ nội dung phải bám sát chủ đề “Lợi ích và rủi ro khi sử dụng AI trong học tập”, thời lượng phù hợp với một bài thuyết trình 10 phút, tránh dùng ngôn ngữ quá phức tạp hoặc quá kỹ thuật. Nội dung cần logic, có tính thuyết phục và truyền cảm hứng.

Yêu cầu 6:

Viết 1 prompt chi tiết yêu cầu AI **sinh ví dụ code minh họa một hàm tính giai thừa**, sau đó yêu cầu AI giải thích cùng một khái niệm ở **3 cấp độ khác nhau**: cho người mới học, cho sinh viên CNTT, và cho kỹ sư phần mềm.

## Prompt:

Bạn là giảng viên CNTT kiêm chuyên gia lập trình. Hãy viết code Java minh họa một hàm tính giai thừa của số nguyên dương n. Code cần ngắn gọn, dễ đọc, không dùng

thư viện ngoài và có thêm comment để giải thích từng phần. Sau khi viết code, hãy giải thích khái niệm “giai thừa” ở ba cấp độ khác nhau. Với người mới học, hãy diễn đạt thật đơn giản, gần gũi với đời sống và tránh thuật ngữ phức tạp. Với sinh viên CNTT, hãy giải thích cách hàm hoạt động, tư duy thuật toán và ý nghĩa cú pháp. Với kỹ sư phần mềm, hãy phân tích chi tiết về độ phức tạp tính toán, so sánh cách cài đặt vòng lặp và đệ quy, đồng thời nhắc đến hướng tối ưu hóa.

Ngữ cảnh của yêu cầu này là một bài học đa cấp độ, giúp cả người mới và người học nâng cao đều có thể tiếp cận và hiểu rõ cùng một khái niệm theo mức độ phù hợp. Mục tiêu là làm rõ kiến thức từ cơ bản đến nâng cao theo ba tầng giải thích.

Bạn phải luôn trình bày code trước rồi mới đến phần giải thích. Ba đoạn giải thích cần viết thành văn bản mạch lạc, không được liệt kê gạch đầu dòng, và mỗi đoạn phải có tiêu đề rõ ràng “Giải thích cho người mới học”, “Giải thích cho sinh viên CNTT” và “Giải thích cho kỹ sư phần mềm”.

Đầu vào của bài toán là một số nguyên dương  $n$ . Đầu ra phải bao gồm code Java tính giai thừa của  $n$ , kèm theo ba phần giải thích ở ba cấp độ theo đúng yêu cầu.

Yêu cầu :

Viết một prompt yêu cầu AI sinh code hàm Python tính tổng các số chẵn từ 1 đến  $n$

## Prompt:

Bạn là giảng viên CNTT. Hãy viết code Python định nghĩa một hàm `sum_even(n)` để tính tổng tất cả các số chẵn từ 1 đến  $n$ . Mục tiêu là giúp sinh viên hiểu cách sử dụng vòng lặp, điều kiện và phép toán cộng trong Python. Ngữ cảnh là một bài học nhập môn lập trình Python dành cho sinh viên năm nhất, nên code cần đơn giản, dễ đọc và dễ hiểu.

Ràng buộc: code không được sử dụng thư viện ngoài, phải có thêm comment giải thích từng bước quan trọng trong hàm. Hàm phải xử lý đúng với mọi giá trị nguyên dương  $n$ . Luôn trình bày code trước, sau đó có một đoạn giải thích ngắn gọn cách hàm hoạt động.

Định dạng: code phải được đặt trong khối python `...`. Phần giải thích viết thành đoạn văn mạch lạc, không dùng gạch đầu dòng.

Input: một số nguyên dương  $n$ .

Output: một đoạn code Python định nghĩa hàm `sum_even(n)` kèm theo phần giải thích rõ ràng.

Yêu cầu:

Viết một prompt yêu cầu AI tìm và giải thích lỗi trong đoạn code sau: (được lưu trữ trong file `array.py` dòng số **17, 18**):

A screenshot of a code editor with a dark background. At the top left, there are three colored circles: red, yellow, and green. Below them, the following Python code is displayed:

```
my_list = [1, 2, 3]
print(my_list[5])
```

**Đáp án:**


**Prompt:**

Bạn là giảng viên CNTT. Hãy tìm và giải thích lỗi trong file `array.py` tại dòng 17–18. Nếu không mở được file thì yêu cầu tôi dán đoạn code. Trình bày đoạn code gốc có số dòng, phân tích lỗi, sửa code trong khối python với comment giải thích, thêm 1–2 test case kiểm chứng và kết thúc bằng phần “Giảng” ngắn rút ra bài học. Tất cả giải thích bằng tiếng Việt, sửa tối thiểu, không thay đổi chức năng.

Output:

Mình đã xem hình bạn gửi. Trong file `array.py`, đoạn code dòng 17–18 như sau (mình thêm số dòng để dễ phân tích):

python

 Copy code

```
17 | my_list = [1, 2, 3]
18 | print(my_list[5])
```


## Phân tích lỗi

- `my_list` có 3 phần tử, chỉ số hợp lệ là `0, 1, 2`.
- Dòng `print(my_list[5])` gọi phần tử chỉ số 5 (thực tế là phần tử thứ 6), nhưng danh sách không có đủ phần tử → gây lỗi `IndexError: list index out of range`.

## Sửa code

Giữ nguyên chức năng là in một phần tử trong list, chỉ cần chọn chỉ số hợp lệ.

python

 Copy code


```
# array.py

# Dòng 17
my_list = [1, 2, 3]

# Dòng 18 - sửa: dùng chỉ số 2 (phần tử cuối) thay vì 5
print(my_list[2]) # In ra phần tử thứ 3 của list, tránh IndexError
```

## Test case kiểm chứng

python

 Copy code

```
# Test 1: danh sách mặc định
my_list = [1, 2, 3]
print(my_list[2]) # Kết quả: 3

# Test 2: danh sách lớn hơn
my_list = [10, 20, 30, 40, 50]
print(my_list[2]) # Kết quả: 30
```

## Giảng

Bài học ở đây: khi truy cập phần tử của list, chỉ số phải nằm trong phạm vi `[0 .. len(list)-1]`. Nếu chỉ số vượt quá, Python sẽ báo lỗi `IndexError`. Vì vậy, hãy luôn kiểm tra độ dài danh sách trước khi truy cập hoặc dùng cú pháp an toàn như `my_list[-1]` để lấy phần tử cuối.

Yêu cầu:

Viết 2 prompt mẫu sẽ nhập trong Cursor để:

- Một prompt yêu cầu Cursor **sinh comment mô tả chi tiết logic của hàm kiểm tra số nguyên tố.**
- Một prompt yêu cầu Cursor **chỉnh sửa toàn bộ code trong project để đổi phong cách đặt tên biến từ snake\_case sang camelCase.**

Đầu dòng 1:

Prompt:

Bạn hãy viết cho tôi 1 hàm có chức năng kiểm tra số nguyên tố bằng javascript, người dùng truyền vào 1 số n, hàm sẽ trả xem số đấy có phải số nguyên tố hay không (true/false), tên hàm viết bằng quy tắc snake case, khi tạo hàm, nhớ thêm các comment chi tiết vào các chức năng để diễn giải chức năng đó bằng JS Doc,

Output:



```

/**
 * Kiểm tra một số có phải là số nguyên tố hay không.
 *
 * Định nghĩa: Số nguyên tố là số nguyên lớn hơn hoặc bằng 2,
 * chỉ có đúng hai ước số dương là 1 và chính nó.
 *
 * Thuật toán:
 * - Loại bỏ các trường hợp không phải số nguyên hoặc nhỏ hơn 2.
 * - Xử lý nhanh cho 2 (là số nguyên tố) và các số chẵn > 2 (không phải nguyên tố).
 * - Duyệt các ước lẻ từ 3 đến căn bậc hai của n. Nếu tồn tại ước chia hết, n không phải nguyên tố.
 *
 * Độ phức tạp:  $O(\sqrt{n})$  cho  $n \geq 2$ .
 *
 * @param {number} n - Số cần kiểm tra. Chấp nhận kiểu number hữu hạn trong JavaScript.
 * @returns {boolean} Trả về true nếu n là số nguyên tố, ngược lại trả về false.
 *
 * @example
 * // true (2 là số nguyên tố)
 * is_prime(2);
 *
 * @example
 * // false (9 = 3 x 3)
 * is_prime(9);
 *
 * @example
 * // false (số âm và số < 2 không phải số nguyên tố)
 * is_prime(-5);
 *
 * @example
 * // false (số không nguyên hoặc NaN/Infinity đều không phải số nguyên tố)
 * is_prime(7.2);
 * is_prime(NaN);
 */
function is_prime(n) {
    // Kiểm tra nhanh: phải là số hữu hạn và là số nguyên
    if (!Number.isFinite(n) || !Number.isInteger(n)) {
        return false;
    }

    // Các số < 2 không phải số nguyên tố
    if (n < 2) {
        return false;
    }

    // 2 là số nguyên tố duy nhất chẵn
    if (n === 2) {
        return true;
    }

    // Loại bỏ số chẵn > 2
    if (n % 2 === 0) {
        return false;
    }

    // Chỉ cần kiểm tra các ước lẻ từ 3 đến sqrt(n)
    const limit = Math.floor(Math.sqrt(n));
    for (let divisor = 3; divisor <= limit; divisor += 2) {
        if (n % divisor === 0) {
            return false;
        }
    }

    // Không tìm thấy ước số nào => n là số nguyên tố
    return true;
}

```

Đầu dòng 2:

Prompt:

Bây giờ tôi muốn chỉnh sửa, tôi muốn đổi code trong project của tôi. chỉnh sửa toàn bộ cách đặt tên biết từ snake\_case sang camelCas

Output:

```
/**
 * Kiểm tra một số có phải là số nguyên tố hay không.
 *
 * Định nghĩa: Số nguyên tố là số nguyên lớn hơn hoặc bằng 2,
 * chỉ có đúng hai ước số dương là 1 và chính nó.
 *
 * Thuật toán:
 * - Loại bỏ các trường hợp không phải số nguyên hoặc nhỏ hơn 2.
 * - Xử lý nhanh cho 2 (là số nguyên tố) và các số chẵn > 2 (không phải nguyên tố).
 * - Duyệt các ước lẻ từ 3 đến căn bậc hai của n. Nếu tồn tại ước chia hết, n không phải nguyên tố.
 *
 * Độ phức tạp:  $O(\sqrt{n})$  cho  $n \geq 2$ .
 *
 * @param {number} n - Số cần kiểm tra. Chấp nhận kiểu number hữu hạn trong JavaScript.
 * @returns {boolean} Trả về true nếu n là số nguyên tố, ngược lại trả về false.
 *
 * @example
 * // true (2 là số nguyên tố)
 * isPrime(2);
 *
 * @example
 * // false (9 = 3 x 3)
 * isPrime(9);
 *
 * @example
 * // false (số âm và số < 2 không phải số nguyên tố)
 * isPrime(-5);
 *
 * @example
 * // false (số không nguyên hoặc NaN/Infinity đều không phải số nguyên tố)
 * isPrime(7.2);
 * isPrime(NaN);
 */
function isPrime(n) {
  // Kiểm tra nhanh: phải là số hữu hạn và là số nguyên
  if (!Number.isFinite(n) || !Number.isInteger(n)) {
    return false;
  }

  // Các số < 2 không phải số nguyên tố
  if (n < 2) {
    return false;
  }

  // 2 là số nguyên tố duy nhất chẵn
  if (n === 2) {
    return true;
  }

  // Loại bỏ số chẵn > 2
  if (n % 2 === 0) {
    return false;
  }

  // Chỉ cần kiểm tra các ước lẻ từ 3 đến sqrt(n)
  const limit = Math.floor(Math.sqrt(n));
  for (let divisor = 3; divisor <= limit; divisor += 2) {
    if (n % divisor === 0) {
      return false;
    }
  }

  // Không tìm thấy ước số nào => n là số nguyên tố
  return true;
}
```