* Viết ra 5 nguyên tắc cá nhân khi sử dụng AI trong học tập lập trình.
* Với mỗi nguyên tắc, kèm ví dụ minh họa ngắn (1–2 câu).  
  Ví dụ: *“Luôn chạy thử lại code do AI sinh ra → tránh lỗi runtime.”*

BÀI LÀM:

**1. Kiểm tra và hiểu rõ code AI tạo ra**

* **Nguyên tắc**: Đừng chỉ sao chép và dán mã nguồn từ AI, hãy kiểm tra và hiểu cách hoạt động của nó.
* **Ví dụ**: Khi AI tạo ra một đoạn mã xử lý dữ liệu, tôi sẽ phân tích các biến và cấu trúc của mã đó trước khi áp dụng vào dự án của mình, để tránh các lỗi tiềm ẩn không dễ nhận thấy.

**2. Chạy thử lại code do AI sinh ra → tránh lỗi runtime**

* **Nguyên tắc**: Sau khi nhận được code từ AI, luôn chạy thử trên môi trường kiểm thử để kiểm tra tính chính xác và hiệu suất.
* **Ví dụ**: AI giúp tôi viết một hàm tìm kiếm nhị phân, nhưng tôi sẽ chạy thử với nhiều bộ dữ liệu khác nhau để đảm bảo hàm hoạt động đúng và không gặp lỗi runtime.

**3. Sử dụng AI để cải thiện, không thay thế hoàn toàn**

* **Nguyên tắc**: AI là công cụ hỗ trợ học tập, không phải công cụ thay thế việc tự học và giải quyết vấn đề.
* **Ví dụ**: Khi học về cấu trúc dữ liệu, tôi dùng AI để tìm hiểu thuật toán mới, nhưng tôi vẫn cố gắng tự viết code để hiểu sâu về cách hoạt động của thuật toán đó.

**4. Kiểm tra hiệu suất và tối ưu hóa code**

* **Nguyên tắc**: Mặc dù AI có thể tạo ra mã nhanh chóng, nhưng bạn cần kiểm tra và tối ưu mã để đạt hiệu suất tốt nhất.
* **Ví dụ**: AI giúp tôi tạo một đoạn mã để xử lý chuỗi, nhưng tôi sẽ chạy benchmark để kiểm tra tốc độ xử lý và tối ưu hóa nếu cần thiết, ví dụ như giảm độ phức tạp thời gian.

**5. Học từ những lỗi và gợi ý của AI**

* **Nguyên tắc**: Khi AI đưa ra gợi ý hoặc tạo mã có lỗi, hãy học từ đó để cải thiện kỹ năng lập trình của mình.
* **Ví dụ**: AI gợi ý sử dụng một phương thức không tối ưu, tôi sẽ tìm hiểu lý do và tự tìm giải pháp thay thế tốt hơn, như chuyển sang một thuật toán tìm kiếm nhanh hơn.