

BÁO CÁO SAU BUỔI THỰC HÀNH

Môn học: Xây dựng giao diện web với HTML và CSS

Tên buổi thực hành: Thực hành HTML Basic, Table, Form

Thời gian thực hành: 180p

Họ và tên sinh viên: Đoàn Mạnh Duy

Mã sinh viên: PTIT-HN-043

Lớp: KS24-CNTT4

Nhóm: 3

I. NỘI DUNG ĐÃ THỰC HÀNH

1. Liệt kê các câu hỏi từ nhóm khác và câu hỏi phản biện

II. CÔNG VIỆC CÁC EM ĐÃ LÀM

1. Công việc cá nhân:

- Viết và chỉnh sửa code Java

2. Công việc nhóm:

Mã yêu cầu: FR-03

Quan hệ:

- Kế thừa GameCharacter
- Thực thi ISkill

Logic nghiệp vụ:

1. Thuộc tính bổ sung:

- mana – Năng lượng phép thuật

2. Tấn công thường attack():

- Nếu mana ≥ 5 :
 - § Gây sát thương = attackPower
 - § Trừ 5 mana
- Nếu mana < 5 :
 - § Gây sát thương = attackPower / 2

3. Chiêu cuối useUltimate():

- Tên chiêu: **“Hỏa cầu”**
- Điều kiện: mana ≥ 50
- Hiệu ứng:
 - § Trừ 50 mana
 - § Gây sát thương lớn (ví dụ: attackPower * 3)

III. KẾT QUẢ CÁC EM ĐẠT ĐƯỢC

Sau buổi thực hành, em đã:

1. Hiểu sâu về bản chất của từ khóa `final` (phân biệt giữa `final` biến và `final` tham chiếu).
2. Nắm vững sự khác biệt giữa `static` và `instance method` trong thiết kế hướng đối tượng.
3. Hiểu rõ quy trình cấp phát bộ nhớ (Stack vs Heap) khi khởi tạo đối tượng

Link mini-project: [link](#)

IV. KHÓ KHĂN VÀ VẤN ĐỀ CÁC EM GẶP PHẢI

1. 1. Khó khăn về kỹ thuật:

- **Lỗi NullPointerException:** Ban đầu khi khởi tạo mảng `Student []` nhưng quên khởi tạo từng phần tử hoặc chưa khởi tạo mảng trong Constructor, dẫn đến lỗi khi gọi phương thức `addStudent`.
 - *Khắc phục:* Đã bổ sung việc khởi tạo mảng `new Student [MAX_SIZE]` ngay trong Constructor của lớp quản lý.
- **Lỗi trôi lệnh (Scanner):** Khi nhập số (`nextInt`) xong chuyển sang nhập chuỗi (`nextLine`) bị bỏ qua bước nhập.
 - *Khắc phục:* Thêm lệnh `sc.nextLine()` để xóa bộ nhớ đệm hoặc chuyển sang dùng `Integer.parseInt(sc.nextLine())` để xử lý triệt để.

2. Khó khăn về tư duy Logic:

- Ban đầu em bị nhầm lẫn rằng biến `final` thì hoàn toàn không thể thay đổi bất cứ thứ gì liên quan đến nó. Sau khi tìm hiểu và phản biện, em đã hiểu rõ sự khác biệt giữa "Thay đổi tham chiếu" (bị cấm) và "Thay đổi trạng thái bên trong" (được phép).
- Việc format bảng hiển thị bằng `printf` ban đầu chưa thẳng hàng do chưa hiểu rõ các tham số `%s`, `%-8s`. Sau khi thử nghiệm nhiều lần em đã căn chỉnh được bảng đẹp mắt.

V. KINH NGHIỆM RÚT RA

1. **Luôn kiểm tra Null:** Trước khi thao tác với mảng đối tượng, luôn phải đảm bảo mảng và các phần tử đã được khởi tạo vùng nhớ (Heap).
2. **Tách biệt trách nhiệm:** Nên tách Class chứa dữ liệu (Student) và Class quản lý nghiệp vụ (StudentManager) riêng biệt thay vì gộp chung để code dễ bảo trì và đúng chuẩn OOP.
3. **Tư duy về bộ nhớ:** Hiểu code chạy trên Stack và Heap như thế nào giúp giải thích được các vấn đề hóc búa (như tại sao final object vẫn sửa được ruột).
4. **Kỹ năng Debug:** Khi gặp lỗi cú pháp hay logic, cần đọc kỹ thông báo lỗi (đặc biệt là dòng báo lỗi đỏ) để khoanh vùng và sửa chữa nhanh chóng.

VI. ĐỀ XUẤT / KIẾN NGHỊ

- Nên có thêm thời gian để thực hành các thuật toán sắp xếp và tìm kiếm nâng cao trên mảng đối tượng.

VII. KẾT LUẬN

1. **Tự đánh giá:** Em đánh giá bản thân đã hiểu khoảng **90%** nội dung bài học, hoàn thành tốt các chức năng được giao và trả lời được các câu hỏi phản biện khó.
2. **Giá trị nhận được:** Buổi thực hành giúp em củng cố vững chắc nền tảng Lập trình hướng đối tượng (OOP), đặc biệt là tư duy về quản lý dữ liệu và bộ nhớ trong Java. Đây là tiền đề quan trọng để em tiếp tục học các kiến thức nâng cao như Kế thừa (Inheritance) và Đa hình (Polymorphism) trong các bài tới.

