Lập trình Python
Crittetr Caretaker với GUI

Xây ứng dụng Critter Caretaker (Chapter 8) cho phép tạo, cho ăn, cho ngủ critter qua giao diện

GIÁO VIÊN HƯỚNG DẪN : TS. Nguyễn Văn Huy

1. NGUYỄN VĂN HOAN MSSV: K225480106023

LỚP : K58KTP.K01

Nội dung chính

- 1.Giới thiệu đề tài
- 2. Giới thiệu hệ thống đã làm gì ,làm như thế nào
- 3.Kết luận:kết quả chạy thục nghiệm như thế nào

1.Giới thiệu đề tài
1.1 Đề tài

- Em thực hiện đề tài "Xây dựng ứng dụng Critter Caretaker với GUI" để áp dụng lập trình hướng đối tượng và Tkinter, tạo ứng dụng chăm sóc critter ảo với giao diện nhập tên, các nút "Tạo", "Cho ăn", "Chơi", "Ngủ", và hiển thị trạng thái (hunger, boredom) theo thời gian thực. Đề tài tập trung thiết kế class Critter và CritterApp, xử lý lỗi khi chưa tạo critter, với mục tiêu nâng cao kỹ năng lập trình và thiết kế giao diện.

Lý do chọn đề tài

- **Giúp học tốt lập trình hướng đối tượng (OOP):**
  Đề tài sử dụng class và object để mô hình hóa một "thú ảo", giúp người học nắm rõ cách định nghĩa thuộc tính và phương thức trong Python.
- **Kết hợp với giao diện người dùng (GUI):**
  Việc xây dựng GUI bằng thư viện Tkinter giúp ứng dụng trở nên trực quan, sinh động và dễ sử dụng, phù hợp với xu hướng phần mềm hiện nay.
- **Tính tương tác và sáng tạo:**
  Người dùng có thể tạo, cho ăn, chơi và chăm sóc Critter, tạo cảm giác như một trò chơi đơn giản nhưng hấp dẫn.
- **Dễ mở rộng và nâng cấp:**
  Đề tài có thể dễ dàng mở rộng thêm các tính năng như nhiều thú, lưu trạng thái, tự động giảm chỉ số theo thời gian, v.v.
- **Phù hợp với sinh viên đang học lập trình cơ bản:**
  Đề tài mang tính thực hành cao, giúp củng cố kiến thức đã học qua việc áp dụng vào sản phẩm thực tế.

2. Giới thiệu hệ thống đã làm gì ,làm như thế nào

- **2.1 Tính năng chính của chương trình**

❖ **Tạo Critter mới**

- Người dùng nhập tên vào ô Entry và nhấn nút **"Tạo"**.

Tạo một đối tượng Critter mới với:

  - Hunger = 0
  - Boredom = 0

- Trạng thái ban đầu sẽ hiển thị ngay.

❖ **Cho ăn Critter**

- Nhấn nút **"Cho ăn"** sẽ:
  - Giảm Hunger (đói) đi 2 đơn vị (min = 0)
  - Nhưng thời gian trôi qua khiến cả Hunger và Boredom tăng 1 trước đó
  - ⇒ Nên thực tế Hunger giảm xuống và Boredom tăng một chút

❖ **Chơi với Critter**

- Nhấn nút **"Chơi"** sẽ:
  - Giảm Boredom (chán) đi 2 đơn vị (min = 0)
  - Nhưng thời gian trôi qua khiến cả Hunger và Boredom tăng 1 trước đó

⇒ Kết quả là Critter bớt chán nhưng có thể đói hơn

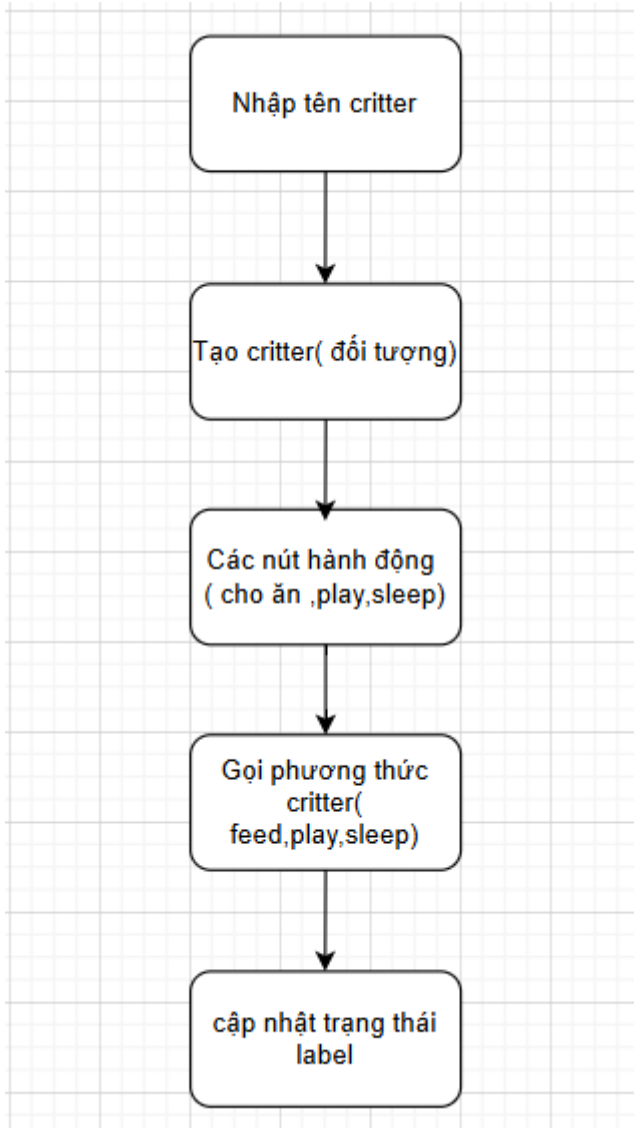
- **Ngủ**
- Nhấn nút **"Ngủ"** sẽ:
  - Tăng Hunger (đói) thêm 1
  - Giảm Boredom (chán) đi 1 (min = 0)
  - Không gọi pass_time() vì ngủ đã là một kiểu "trôi thời gian" đặc biệt

- **Cập nhật trạng thái Critter theo thời gian thực**
- Mỗi khi bạn tương tác (tạo, cho ăn, chơi, ngủ), trạng thái hiện tại của Critter được cập nhật và hiển thị trên Label:
  - Ví dụ: Bob - Hunger: 2, Boredom: 1

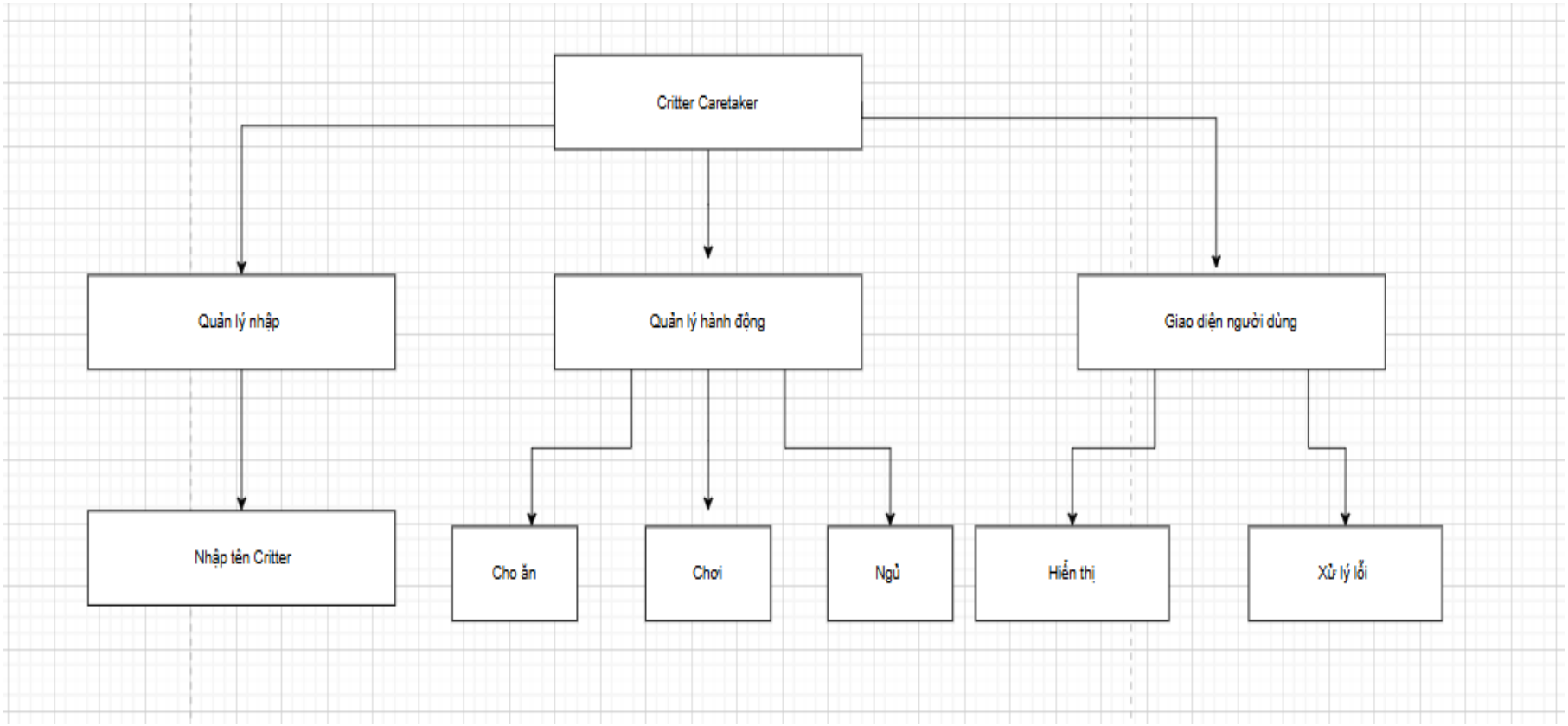
- **2.2.Quy trình hoạt động tổng quát của chương trình**

Nhập tên và tạo Critter
→ Người dùng nhập tên, nhấn "Tạo" → khởi tạo đối tượng Critter mới.
Chọn hành động
→ Người dùng bấm các nút: "Cho ăn", "Chơi", hoặc "Ngủ".
Cập nhật trạng thái Critter
→ Mỗi hành động thay đổi hunger/boredom → Label hiển thị trạng thái mới.
Kiểm tra lỗi
→ Nếu chưa tạo Critter mà bấm nút → hiện thông báo lỗi.

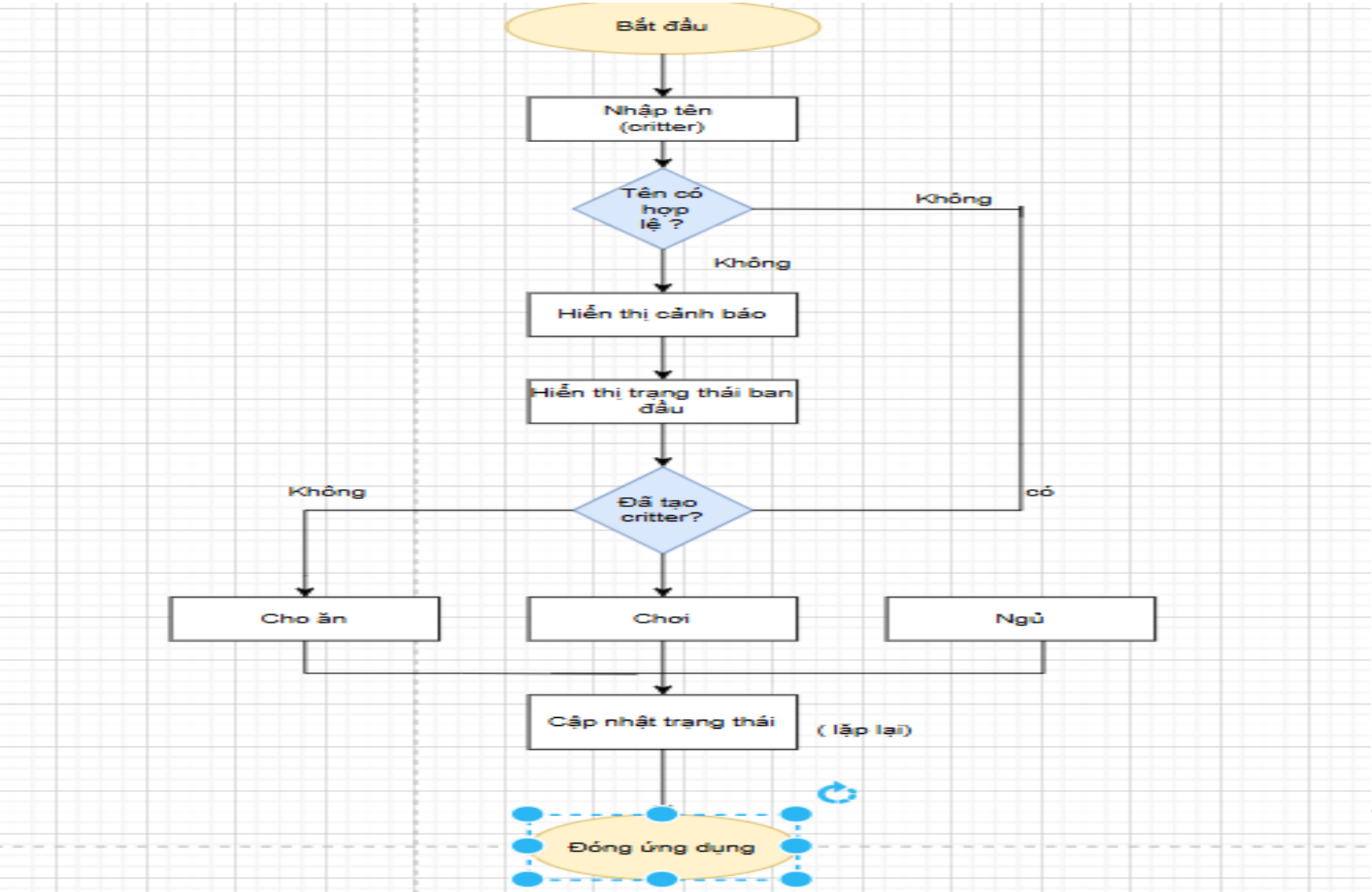
2.3. Sơ đồ khối hệ thống



2.4.Biểu đồ phân cấp chức năng



24.Sơ đồ các thuật toán chính



2.6. Cấu trúc dữ liệu

1. Lớp Critter

Đây là lớp chính dùng để lưu trữ trạng thái và hành vi của thú cưng ảo.

❖ **Thuộc tính (Attributes):**

- name: tên Critter (chuỗi do người dùng nhập).
- hunger: mức độ đói (kiểu số nguyên).
- boredom: mức độ chán (kiểu số nguyên).

❖ **Phương thức (Methods):**

- __init__(self, name): khởi tạo Critter với hunger = 0, boredom = 0.
- feed(self): giảm hunger.
- play(self): giảm boredom.
- sleep(self): giảm cả hunger và boredom nhẹ.
- __str__(self): trả về chuỗi mô tả trạng thái hiện tại.

2. Biến toàn cục (trong GUI):

- current_critter: Biến dùng để lưu Critter đang hoạt động (hoặc None nếu chưa tạo).

3. Widgets giao diện (Tkinter):

Các thành phần giao diện lưu trữ và xử lý dữ liệu đầu vào và đầu ra:

- Entry: nhập tên Critter.
- Label: hiển thị trạng thái Critter.
- Button: thực hiện các hành động (Tạo, Cho ăn, Chơi, Ngủ).

3.Kết luận:kết quả chạy thục nghiệm như thế nào

4.1.Thực nghiệm

Chương trình đã được chạy thử và kiểm tra các tính năng chính:

Giao diện:

Tạo Critter:

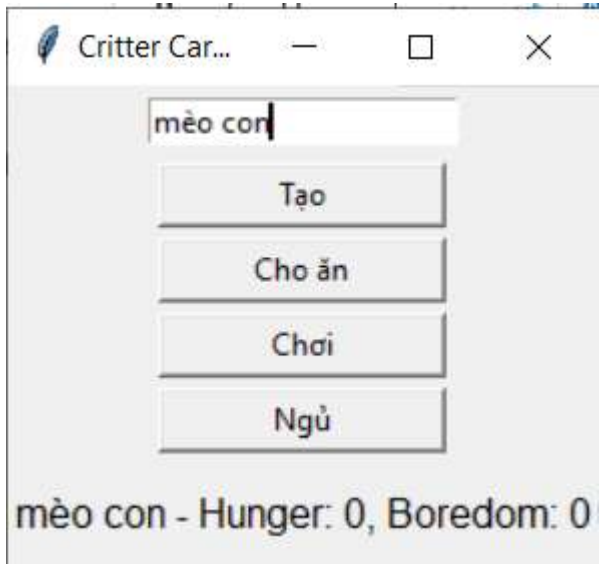
Nhập tên thú ảo vào ô trắng trên cùng.

Nhấn nút “Tạo” để khởi tạo thú cưng mới.

Sau khi tạo, thú cưng bắt đầu với:

Hunger (đói) = 0

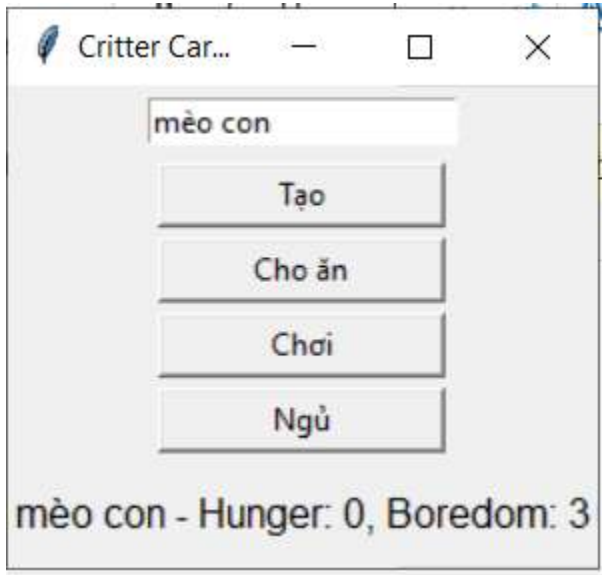
Boredom (chán) = 0



Cho ăn:

Nhấn nút “Cho ăn” để giảm mức đói (hunger).

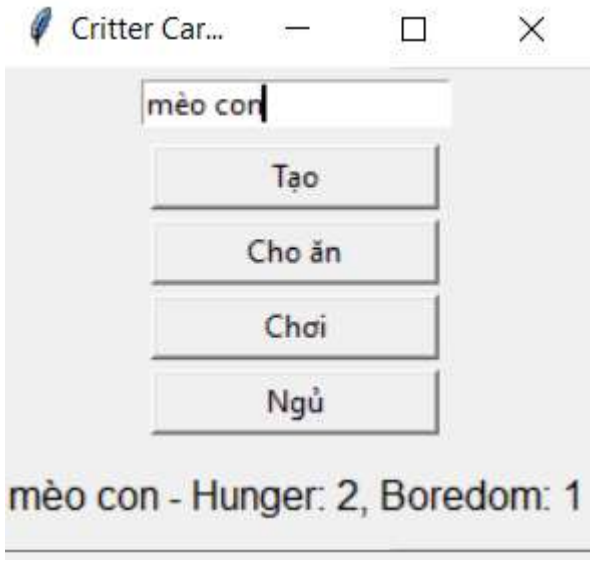
Mỗi lần ăn, mức độ đói giảm, thú khỏe hơn.



Chơi:

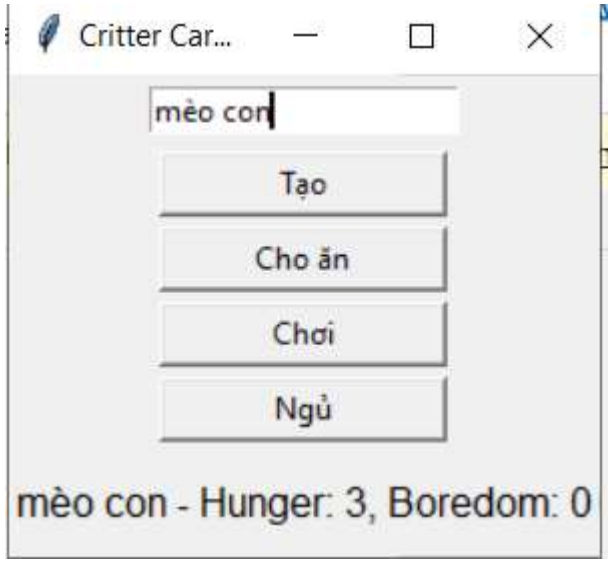
Nhấn nút “Chơi” để giảm mức chán (boredom).

Càng chơi nhiều, thú càng vui vẻ.



Ngủ:

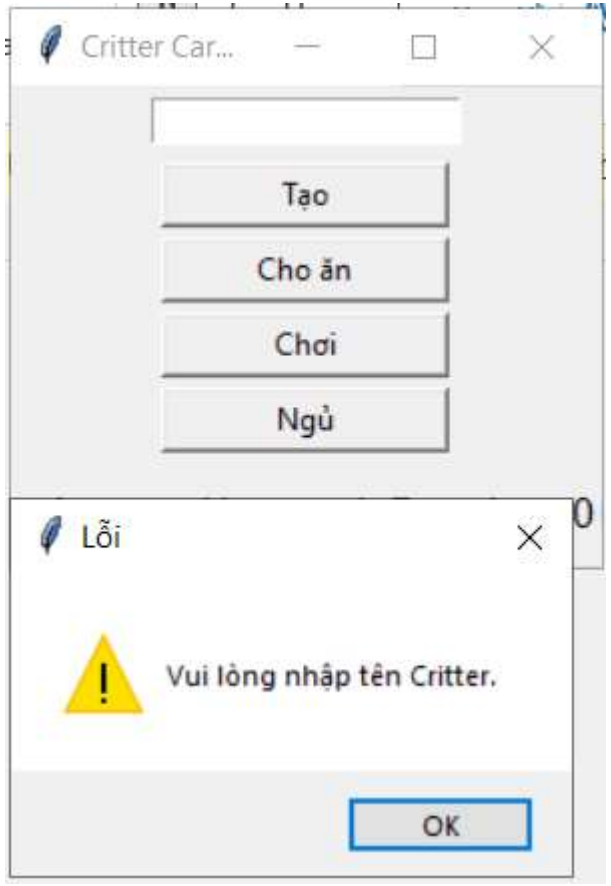
Nhấn nút “Ngủ” để hồi phục cả hai trạng thái một cách nhẹ nhàng.



Cảnh báo và lỗi:

Nếu chưa tạo Critter, mà nhấn các nút hành động, chương trình sẽ hiện thông báo “Chưa có Critter nào.”

Sau mỗi hành động, trạng thái hiện tại của Critter được hiển thị dưới cùng.



Mục tiêu:

Giữ cho thú cưng của bạn luôn khỏe mạnh và vui vẻ bằng cách thường xuyên cho ăn và chơi.

3.1 Kết luận

Chương trình **Critter Caretaker** là một ứng dụng đơn giản nhưng hiệu quả trong việc minh họa các kiến thức lập trình hướng đối tượng kết hợp giao diện đồ họa (GUI) với Tkinter. Người dùng có thể tương tác với một thú cưng ảo thông qua các hành động như **tạo, cho ăn, chơi, ngủ**, từ đó cập nhật và theo dõi trạng thái đói và chán của Critter.

Thông qua quá trình xây dựng chương trình, sinh viên rèn luyện được:

Tư duy thiết kế **class**, phương thức và thuộc tính.

Cách **xử lý sự kiện** với các nút lệnh trong GUI.

Kỹ năng **kết hợp logic xử lý và hiển thị kết quả** theo thời gian thực.

Cách **kiểm soát lỗi người dùng** (chưa tạo Critter đã thao tác).

Chương trình không chỉ mang tính học thuật mà còn có yếu tố thú vị, giúp người học dễ tiếp cận và ghi nhớ kiến thức tốt hơn.