TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP

KHOA ĐIỆN TỬ

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

BÀI TẬP KẾT THÚC MÔN HỌC
LẬP TRÌNH PYTHON

SINH VIÊN	**:**	**NGUYỄN VĂN HOAN**
LỚP	**:**	**K58KTP**
GIÁO VIÊN GIẢNG DẠY	**:**	**TS.NGUYỄN VĂN HUY**
Link GitHub	**:**	https://github.com/Hoan556/-BTL_Ke_thuc_mon_hoc.PyThon

THÁI NGUYÊN-2025

TRƯỜNG ĐHKTCN **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

KHOA ĐIỆN TỬ ***Độc lập - Tự do - Hạnh phúc***

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC: LẬP TRÌNH PYTHON

BỘ MÔN:CÔNG NGHỆ THÔNG TIN

Sinhviên: Nguyễn Văn Hoan MSSV: K225480106023

Lớp: K58KTP.K01 **Ngành:** Kỹ thuật máy tính

Giáo viên hướng dẫn: TS. Nguyễn Văn Huy

Ngày giao đề: 20/05/2025 **Ngày hoàn thành:** 10/06/2025

Tên đề tài: Critter Caretaker với GUI

Bài 4. Critter Caretaker với GUI

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

Đầu vào – đầu ra:

- Đầu vào: Tên critter từ Entry, các nút hành động.
- Đầu ra: Trạng thái critter (hunger, boredom) hiển thị trên Label.

Tính năng yêu cầu:

- Class Critter với attributes và methods.
- GUI: Entry tạo critter, Buttons: “Tạo”, “Cho ăn”, “Chơi”, “Ngủ”.
- Cập nhật trạng thái real-time.
- Bắt lỗi khi chưa tạo critter.

Kiểm tra & kết quả mẫu:

- Tạo “Bob” → Hunger=0, Boredom=0.
- Nhấn “Cho ăn” → Hunger giảm, Label cập nhật.

Các bước triển khai:

1. Thiết kế class Critter theo sách.
2. Tạo CritterApp class mở cửa sổ, khởi tạo widgets.

Map từng button tới method tương ứng của Critter, call display().

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

……
……
……

……

Xếp loại : …… Điểm : ……

Thái Nguyên, ngày 9 tháng 6 năm 2025.

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

LỜI CAM ĐOAN

Em xin cam đoan rằng báo cáo này là kết quả làm việc độc lập của bản thân, dựa trên các kiến thức và kỹ năng đã học. Mọi thông tin, số liệu, và nội dung được trình bày trong báo cáo đều được thực hiện một cách trung thực và không sao chép từ bất kỳ nguồn nào nếu không được trích dẫn rõ ràng. Em chịu hoàn toàn trách nhiệm về tính chính xác và trung thực của báo cáo này.

Em cũng xin cam kết rằng mã nguồn và ứng dụng máy tính GUI được phát triển trong báo cáo là do em tự viết và hoàn thiện, dựa trên yêu cầu đề bài, với sự hỗ trợ từ các tài liệu học tập và công cụ lập trình hợp lệ.

Thái Nguyên. ngày 09 tháng 06 năm 2025

Sinh viên thực hiên

(*Ký và ghi rõ họ tên)*

MỤC LỤC

LỜI MỞ ĐẦU

Trong quá trình học tập về lập trình hướng đối tượng và phát triển giao diện người dùng, em đã có cơ hội khám phá ngôn ngữ Python cùng thư viện Tkinter – một công cụ hữu ích để tạo ra các ứng dụng giao diện đồ họa (GUI) đơn giản nhưng hiệu quả. Đề bài "Xây dựng ứng dụng Critter Caretaker" không chỉ giúp em củng cố kiến thức về quản lý đối tượng và xử lý sự kiện, mà còn rèn luyện kỹ năng thiết kế giao diện thân thiện, cập nhật trạng thái theo thời gian thực, và xử lý các trường hợp lỗi.

Báo cáo này được thực hiện với mục tiêu xây dựng một ứng dụng cho phép em tạo, chăm sóc (cho ăn, chơi, cho ngủ) một critter ảo thông qua giao diện GUI. Ứng dụng bao gồm class Critter để quản lý trạng thái (hunger, boredom) và class CritterApp để tạo giao diện, kết nối các nút hành động với phương thức tương ứng, đồng thời đảm bảo cập nhật trạng thái ngay lập tức. Quá trình thực hiện giúp em hiểu rõ hơn về cách áp dụng lý thuyết vào thực tiễn, đặc biệt trong việc kiểm tra lỗi và tối ưu hóa trải nghiệm người dùng.

Em xin gửi lời cảm ơn chân thành đến thầy TS.Nguyễn Văn Huy đã hướng dẫn, hỗ trợ em trong quá trình thực hiện đề tài này. Dù đã cố gắng hết sức, báo cáo vẫn có thể còn những thiếu sót. Em rất mong nhận được những ý kiến đóng góp quý báu để hoàn thiện hơn trong tương lai.

Thái Nguyên, ngày 09 tháng 06 năm 2025,

CHƯƠNG 1.GIỚI THIỆU ĐẦU BÀI

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.

1.2.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

1.3.Thách thức khi thực hiện

1.Quản lý trạng thái critter.
Cần xử lý hợp lý giữa hành động (cho ăn, chơi...) và việc thời gian trôi (tăng hunger/boredom).

2.Đồng bộ GUI và logic.
Kết nối nút bấm với phương thức class và cập nhật giao diện kịp thời.

3.Xử lý lỗi người dùng.
Tránh crash khi người dùng nhấn nút mà chưa tạo critter.

4.Giao diện đơn giản nhưng rõ ràng.
Bố trí Entry, Button, Label hợp lý và dễ dùng.

1.4.Kiến thức được vận dụng

❖ **Lập trình hướng đối tượng**

Tạo class Critter với thuộc tính và phương thức.

❖ **Xử lý sự kiện trong GUI**

Dùng tkinter để tạo nút, nhập liệu, cập nhật giao diện.

❖ **Điều kiện và kiểm tra lỗi**

Kiểm tra đầu vào, xử lý khi chưa tạo critter.

❖ **Cập nhật trạng thái theo hành vi**

Gắn logic thay đổi trạng thái vào các nút hành động.

Tư duy thiết kế hệ thống đơn giản với giao diện người dùng và xử lý sự kiện.

CHƯƠNG 2.CƠ SỞ LÝ THUYẾT

2.1. Ngôn ngữ Python

2.1.1. Giới thiệu

Python là một ngôn ngữ lập trình thông dịch (interpreted), đa mục đích, hướng đối tượng (object-oriented) và bậc cao (high-level) với ngữ nghĩa động (dynamic semantics). Python hỗ trợ khái niệm module và package, khuyến khích tái sử dụng mã và phát triển theo hướng mô-đun. Bộ thông dịch cùng thư viện chuẩn được phát hành miễn phí dưới dạng mã nguồn lẫn nhị phân trên hầu hết các nền tảng phổ biến và có thể phân phối tự do.

Một số đặc điểm nổi bật:

- Kết nối dễ dàng với môi trường khác: tích hợp COM, .NET (IronPython), JVM (Jython), chia sẻ thư viện C/C++ (CPython) hoặc giao tiếp qua ICE, CORBA…
- Đa nền tảng: chạy nhất quán trên Windows, Linux/Unix, macOS, Android, iOS; cũng tồn tại các bản chạy trên .NET, JVM, thiết bị di động.
- Cú pháp đơn giản, dễ học; cộng đồng lớn và hệ sinh thái thư viện phong phú.
- Mã nguồn mở: tuân theo giấy phép của Python Software Foundation, cho phép sử dụng trong cả môi trường thương mại.

2.2. Thư viện Tkinter

2.2.1.Tkinter là gì?

Tkinter là thư viện tiêu chuẩn đi kèm với Python, hỗ trợ xây dựng các ứng dụng có giao diện đồ họa. Nó giúp tạo ra các thành phần như ô nhập liệu (Entry), nút bấm (Button), nhãn hiển thị (Label), lựa chọn (Radiobutton), hộp thoại thông báo (messagebox)…

Các thành phần thường dùng:

- Tk(): Tạo cửa sổ chính cho ứng dụng.
- Label: Hiển thị văn bản trên giao diện.
- Entry: Nhập dữ liệu từ người dùng.
- Button: Thực hiện hành động khi được nhấn.
- Radiobutton: Cho phép chọn một trong nhiều phép toán.
- messagebox.showerror(): Hiển thị thông báo lỗi.

Hệ thống bố cục grid() của Tkinter cho phép chia layout thành lưới (row, column), giúp dễ bố trí giao diện.

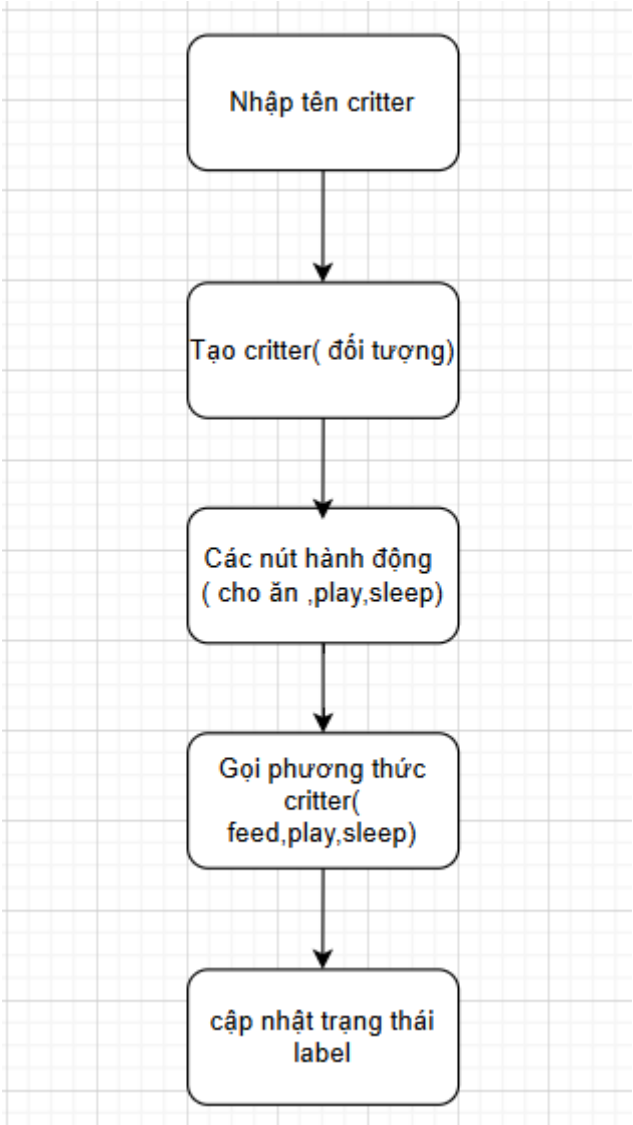
2.3.Quy trình hoạt động tổng quát của chương trình

- 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.

CHƯƠNG 3. THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

3.1. Sơ đồ khối hệ thống

Dưới đây là sơ đồ khối mô tả các module chính trong chương trình:



1. Nhập tên Critter (Entry widget)

Người dùng gõ tên Critter vào ô nhập liệu.

Đây là bước khởi tạo thông tin ban đầu cho một sinh vật.

---

2. Tạo Critter (Khởi tạo đối tượng)

Khi nhấn nút "Tạo", chương trình tạo một đối tượng Critter mới dựa trên tên nhập vào.

Gán giá trị mặc định cho hunger và boredom (thường là 0).

3. Các nút hành động (Cho ăn, Chơi, Ngủ)

Các nút này cho phép người dùng thực hiện hành động chăm sóc Critter.

Mỗi nút tương ứng với một phương thức xử lý hành vi.

4. Gọi phương thức Critter

Khi người dùng nhấn nút:

Gọi hàm tương ứng trong class Critter (feed(), play(), sleep()).

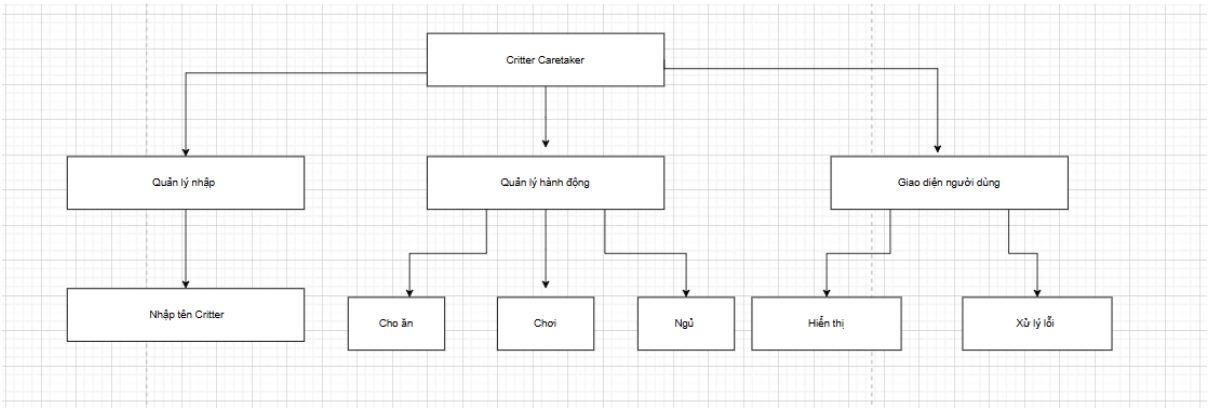
Các hàm này thay đổi trạng thái của Critter (giảm hunger hoặc boredom).

5. Cập nhật trạng thái (Hiển thị trên Label)

Sau mỗi hành động, giao diện cập nhật trạng thái mới của Critter (hunger, boredom).

Dữ liệu hiển thị ở dạng chuỗiTên - Hunger: x, Boredom: y

3.1.1.Biểu đồ phân cấp chức năng



- – Chức năng tổng thể

Critter Caretaker là chức năng tổng quát của chương trình, chịu trách nhiệm tạo và quản lý sinh vật (Critter) thông qua giao diện người dùng.

- – Các chức năng chính

Quản lý nhập dữ liệu

Cho phép người dùng nhập tên Critter qua ô nhập liệu (Entry).

Quản lý hành động Critter

Bao gồm các chức năng chăm sóc Critter:

Cho ăn: Giảm độ đói.

Chơi: Giảm độ chán.

Ngủ: Hồi phục Critter.

Giao diện người dùng

Gồm các thành phần hiển thị và tương tác:

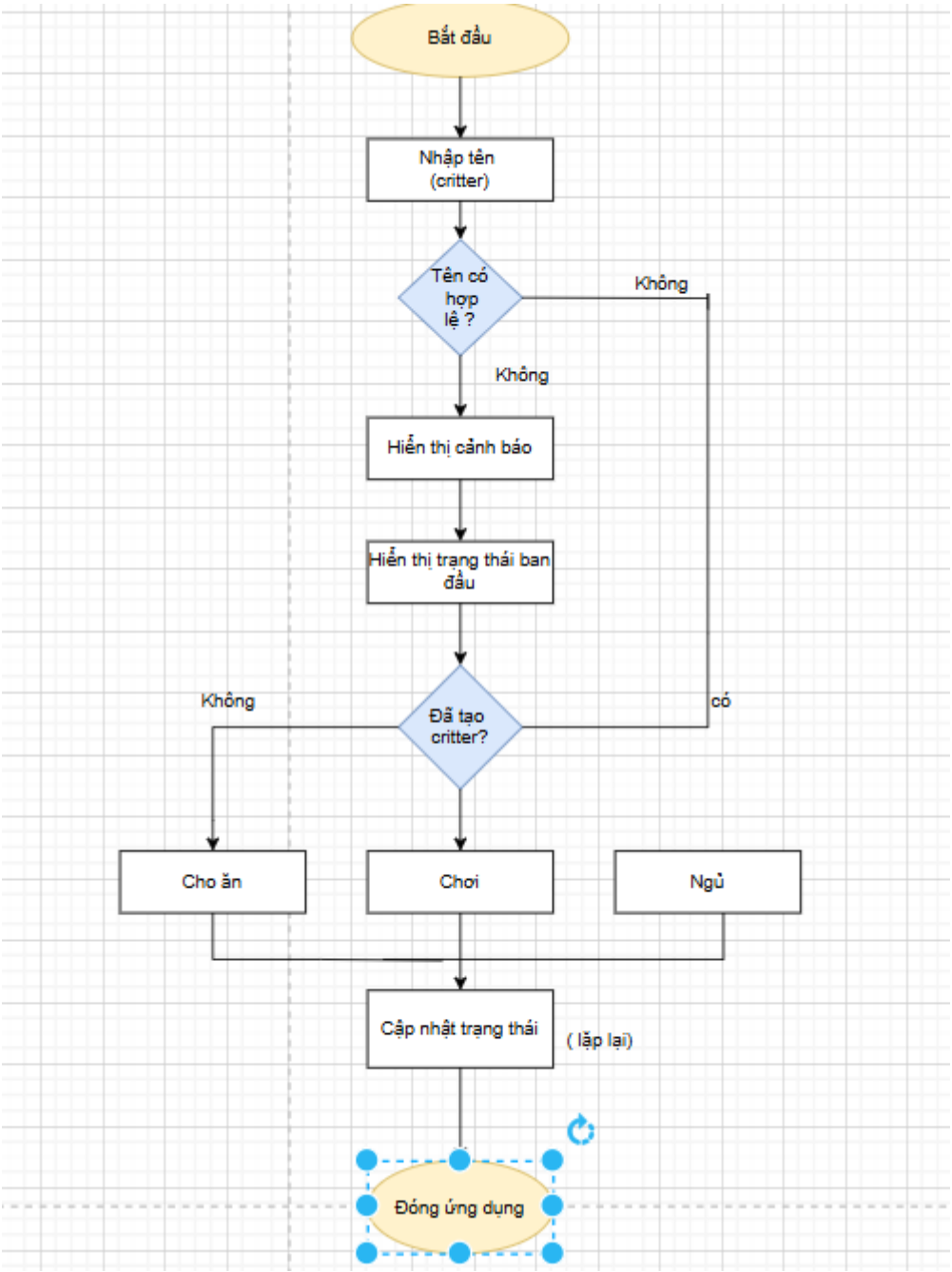
Hiển thị trạng thái: Cập nhật hunger và boredom.

Xử lý lỗi: Cảnh báo nếu người dùng chưa tạo Critter.

- – Chức năng con

Mỗi chức năng chính lại chia nhỏ thành các chức năng con cụ thể, đảm bảo hệ thống hoạt động logic, rõ ràng và dễ bảo trì.

3.2.Sơ đồ các thuật toán chính



Các thuật toán chính có trong Critter Caretaker GUI

---

1. Nhập tên Critter

Chức năng: Cho phép người dùng tạo một Critter bằng cách nhập tên.

Đầu vào:

Tên từ ô nhập (name_entry)

Xử lý:

Đọc giá trị từ ô nhập.

Kiểm tra xem tên có rỗng không.

Đầu ra:

Nếu hợp lệ: tạo đối tượng Critter.

Nếu không: hiển thị cảnh báo.

---

2. Cho ăn (Feed)

Chức năng: Cho Critter ăn để giảm đói.

Đầu vào:

Nút "Cho ăn" được nhấn.

Xử lý:

Kiểm tra đã tạo Critter chưa.

Gọi phương thức feed():

Tăng đói và buồn chán (qua pass_time()).

Giảm đói 2 đơn vị.

Đầu ra:

Cập nhật trạng thái trên giao diện hoặc hiển thị lỗi nếu chưa tạo Critter.

---

3. Chơi (Play)

Chức năng: Cho Critter chơi để giảm buồn chán.

Đầu vào:

Nút "Chơi" được nhấn.

Xử lý:

Kiểm tra đã tạo Critter chưa.

Gọi phương thức play():

Tăng đói và buồn chán (qua pass_time()).

Giảm buồn chán 2 đơn vị.

Đầu ra:

Cập nhật trạng thái hoặc báo lỗi nếu chưa có Critter.

---

4. Ngủ (Sleep)

Chức năng: Cho Critter nghỉ để hồi phục một chút tâm trạng.

Đầu vào:

Nút "Ngủ" được nhấn.

Xử lý:

Kiểm tra đã tạo Critter chưa.

Gọi phương thức sleep():

Tăng đói 1 đơn vị.

Giảm buồn chán 1 đơn vị.

Đầu ra:

Trạng thái mới được hiển thị hoặc báo lỗi.

---

5. Cập nhật trạng thái Critter

Chức năng: Hiển thị trạng thái mới của Critter sau mỗi hành động.

Đầu vào:

Thuộc tính hunger, boredom của Critter.

Xử lý:

Gán chuỗi trạng thái vào status_label.

Đầu ra:

Dòng chữ: "Tên - Hunger: x, Boredom: y" trên giao diện.

---

6. Kiểm tra đối tượng Critter

Chức năng: Đảm bảo người dùng đã tạo Critter trước khi tương tác.

Đầu vào:

Hành động từ các nút (ăn, chơi, ngủ).

Xử lý:

Kiểm tra self.critter có tồn tại không.

Đầu ra:

Tiếp tục thực hiện nếu có Critter, hoặc hiển thị lỗi.

3.3. 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:

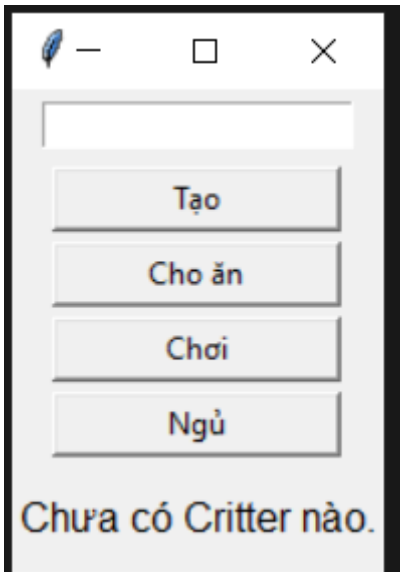
- Entry: nhập tên Critter.
- Label: hiển thị trạng thái Critter.
- Button: thực hiện các hành động (Tạo, Cho ăn, Chơi, Ngủ).

CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT LUẬN

4.1.Thực nghiệm

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

- Giao diện:



❖ **Tạo Critter:**

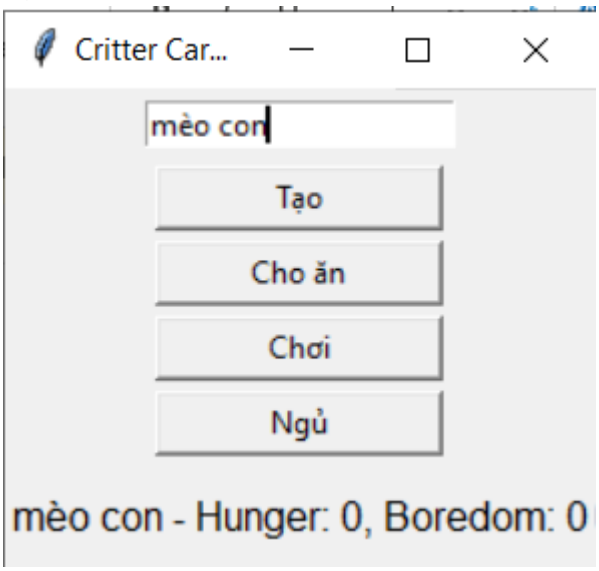
Nhập tên thú ảo vào ô trắng trên cùng.

Nhấn nút "Tạo" để khởi tạo thú cưng mới.

Sau khi tạo, thú cưng bắt đầu với:

Hunger (đói) = 0

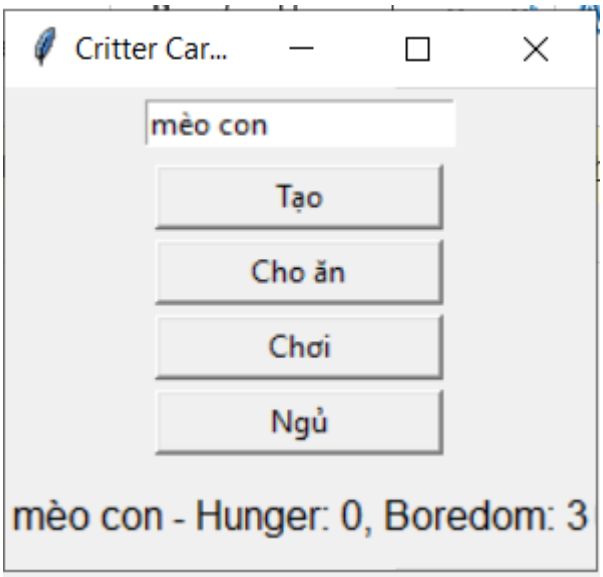
Boredom (chán) = 0



❖ **Cho ăn**:

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

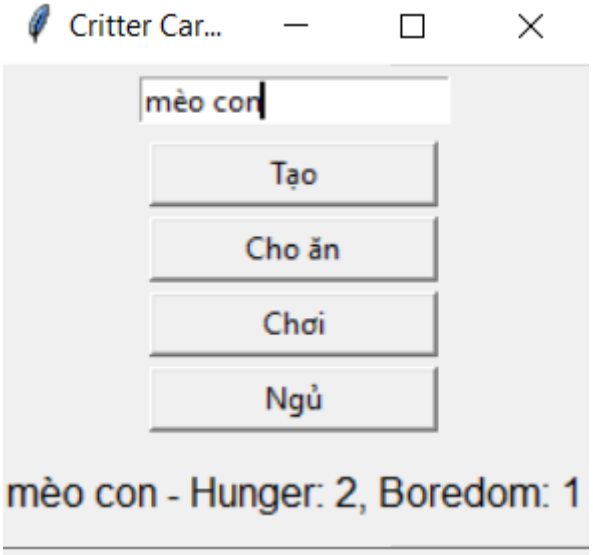
Mỗi lần ăn, mức độ đói giảm, thú khỏe hơn.



❖ **Chơi:**

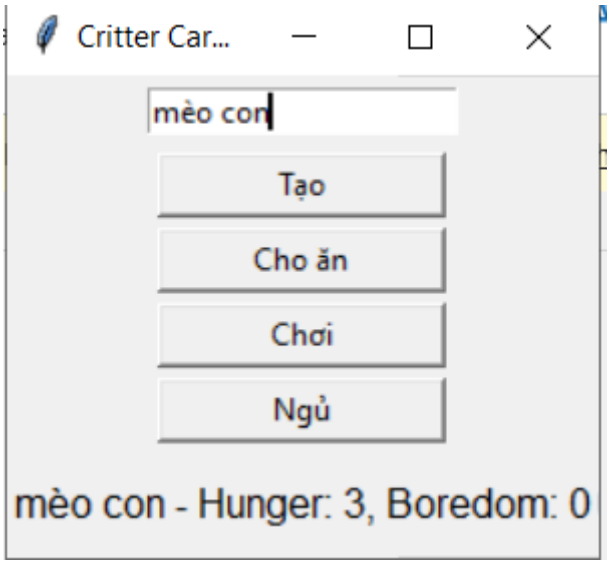
Nhấn nút "Chơi" để giảm mức chán (boredom).

Càng chơi nhiều, thú càng vui vẻ.



❖ **Ngủ:**

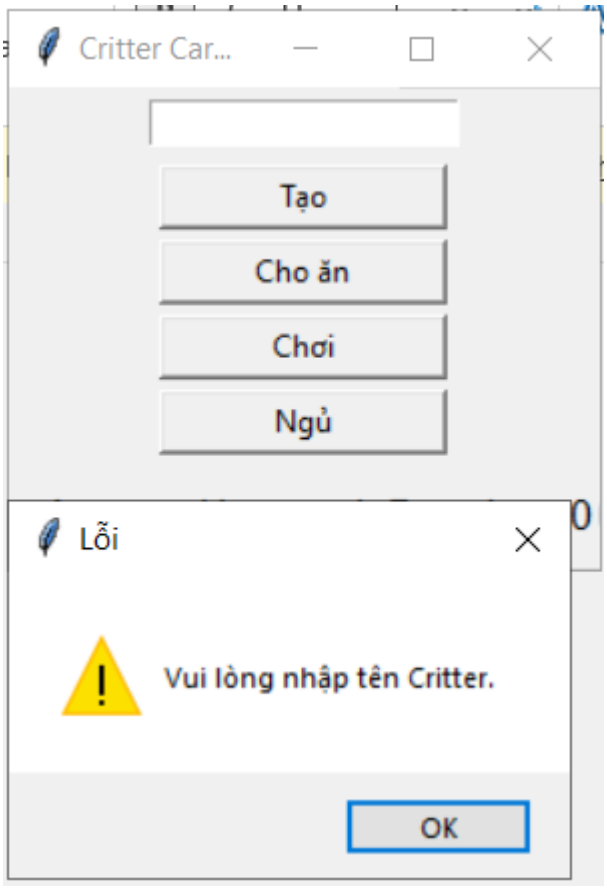
Nhấn nút "Ngủ" để hồi phục cả hai trạng thái một cách nhẹ nhàng.



❖ **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.

4.2.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.