TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP
KHOA ĐIỆN TỬ
Bộ môn: Công nghệ Thông tin.

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC

LẬP TRÌNH PYTHON

Sinh viên: Dương Quang Minh

Lớp: K58KTP.K01

Giáo viên GIẢNG DẠY: Nguyễn Văn Huy

Link GitHub: https://github.com/D-Q-Minh/btl-lap-trinh-python

Thái Nguyên – 2025

TRƯỜNG ĐHKTCN
KHOA ĐIỆN TỬ

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC: LẬP TRÌNH PYTHON

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

Sinh viên: Dương Quang Minh

Lớp: K58KTP.K01 *Ngành: Kỹ Thuật Máy Tính*

Giáo viên hướng dẫn: Nguyễn Văn Huy

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

Tên đề tài : Xây game Astrocrash (Chapter 12) với pygame: điều khiển tàu, bắn asteroid, âm thanh.

Yêu cầu :

Đầu vào – đầu ra:

- Đầu vào: Phím mũi tên quay/move, phím cách bắn.
- Đầu ra: Điểm, hiệu ứng nổ, nhạc nền.

Tính năng yêu cầu:

- Quay sprite, di chuyển, bắn tên lửa (Missile).
- Collisions, di chuyển asteroid.
- Phát sound và music.

Kiểm tra & kết quả mẫu:

Bắn trúng asteroid → nổ +10 điểm, có sound “boom”.

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày….tháng…..năm 20....

GIÁO VIÊN HƯỚNG DẪN

(Ký ghi rõ họ tên)

MỤC LỤC

LỜI NÓI ĐẦU

Asteroids là một game arcade cổ điển. Nội dung game là điều khiển tàu vũ trụ tránh né và bắn các tiểu hành tinh để tăng số điểm. Bài tập lớn này nhằm xây dựng game Asteroids với ngôn ngữ lập trình Python.

CHƯƠNG 1: GIỚI THIỆU ĐẦU BÀI

1.1. Giới thiệu đề tài

Bài tập số 8 yêu cầu xây game Astrocrash (Chapter 12) với pygame: điều khiển tàu, bắn asteroid, âm thanh.

Đầu vào – đầu ra:

- Đầu vào: Phím mũi tên quay/move, phím cách bắn.
- Đầu ra: Điểm, hiệu ứng nổ, nhạc nền.

Tính năng yêu cầu:

- Quay sprite, di chuyển, bắn tên lửa (Missile).
- Collisions, di chuyển asteroid.
- Phát sound và music.

Kiểm tra & kết quả mẫu:

- Bắn trúng asteroid → nổ +10 điểm, có sound "boom".

1.2. Các tính năng chính của chương trình

- Di chuyển tàu: tăng, giảm tốc độ tàu; xoay hướng tàu.
- Bắn: tàu bắn tên lửa, tên lửa chạm asteroid thì asteroid bị phá hủy
- Tính điểm: tăng 10 điểm khi 1 asteroid bị phá hủy

1.3. Kiến thức được áp dụng

- Lập trình hướng đối tượng: Lập trình hướng đối tượng (OOP) là một mô hình lập trình dựa trên khái niệm về "đối tượng", có thể chứa dữ liệu (thuộc tính) và mã (phương thức). Trong Python, OOP là một phần cốt lõi và được sử dụng rộng rãi để xây dựng các ứng dụng có cấu trúc, dễ bảo trì và mở rộng.
- Spirite: Trong Pygame, Sprite là một khái niệm rất quan trọng, dùng để biểu diễn bất kỳ đối tượng đồ họa nào trong game. Một Sprite trong Pygame là một lớp (class) được thiết kế để dễ dàng quản lý hình ảnh và vị trí của các đối tượng trên màn hình.
- Vòng lặp game (Game Loop): Game loop là một vòng lặp vô tận, liên tục chạy và thực hiện các tác vụ cần thiết để cập nhật trạng thái của game và vẽ mọi thứ lên màn hình.
    - Xử lý sự kiện (Event Handling): Game loop sẽ kiểm tra xem có bất kỳ sự kiện nào xảy ra không và phản ứng lại.
    - Cập nhật trạng thái game (Game State Update): Sau khi xử lý các sự kiện đầu vào, game loop sẽ cập nhật trạng thái của tất cả các đối tượng trong game

- Vẽ/Hiển thị (Drawing/Rendering): Sau khi tất cả các đối tượng đã được cập nhật trạng thái, game loop sẽ vẽ lại toàn bộ khung hình lên màn hình.
- Kiểm soát tốc độ khung hình (Frame Rate Control): game loop thường có một cơ chế để kiểm soát tốc độ khung hình (FPS - Frames Per Second). Sử dụng pygame.time.Clock().tick(FPS) để giới hạn số lượng frame tối đa mỗi giây, giúp game không chạy quá nhanh hoặc tiêu tốn quá nhiều tài nguyên CPU.

CHƯƠNG 2: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

2.1. Sơ đồ khối hệ thống

Chương trình thiết kế gồm:

- Phần giao diện: hiển thị màn hình game, số điểm, vị trí của ship, missile, asteroid
- Phần xử lý dữ liệu: xử lý dữ liệu từ bàn phím, cập nhật dữ liệu vị trí

2.2. Sơ đồ khối các thuật toán chính

2.2.1. Thuật toán điều khiển tàu

- Đầu vào: Các phím mũi tên
- Xử lý:
  - Nhấn phím mũi tên lên thì tàu tăng tốc, tiến lên
  - Nhấn phím mũi tên xuống thì tàu giảm tốc, lùi lại
  - Nhấn phím mũi trên trái thì tàu xoay hướng tàu sang trái
  - Nhấn phím mũi trên phải thì tàu xoay hướng tàu sang phải
- Đầu ra: Cập nhật vị trí và hướng mới của tàu

2.2.2. Thuật toán bắn tên lửa

- Đầu vào: Phím cách, vị trí và hướng của tàu
- Xử lý:
  - Xác định vị trí đầu tàu
  - Tạo đối tượng missile tại vị trí tàu
- Đầu ra: đối tượng missile bay thẳng theo hướng đầu tàu

2.2.3. Thuật toán di chuyển asteroid

- Đầu vào: Vị trí ban đầu (mép màn hình), hướng và tốc độ ngẫu nhiên
- Xử lý:
  - Cập nhật vị trí mỗi khung hình (dựa trên góc và tốc độ)
  - Khi ra ngoài rìa, vòng lại từ phía đối diện
- Đầu ra: Vị trí asteroid được cập nhật liên tục và di chuyển ngẫu nhiên.

2.2.4. Thuật toán va chạm (Collision Detection)

2.2.4.1. Missile và asteroid

- Đầu vào: Danh sách missiles và asteroids
- Xử lý:

- Sử dụng pygame.sprite.groupcollide() để kiểm tra va chạm
- Khi va chạm:
    - Xóa asteroid và missile khỏi màn hình
    - Tăng điểm
    - Tạo hiệu ứng Explosion tại vị trí va chạm

- Đầu ra: Cập nhật điểm số, phát âm thanh nổ, hiệu ứng nổ

2.2.4.2. tàu và asteroid

- Đầu vào: tàu và asteroid
- Xử lý:
    - Sử dụng pygame.sprite.spritecollideany() để kiểm tra va chạm
    - Nếu va chạm: kết thúc game
- Đầu ra: Kết thúc game, hiện thông báo "Game Over!"

2.2.5. Thuật toán hiển thị màn hình

- Đầu vào: Vị trí và hình ảnh của tất cả đối tượng (ship, missile, asteroid, explosion)
- Xử lý:
    - Hiển thị nền, sau đó từng sprite lên màn hình
    - Hiển thị điểm số (score)
- Đầu ra: Khung cửa sổ game hoàn chỉnh

2.2.6. Thuật toán bắt đầu game khi nhấn phím

- Đầu vào: Trạng thái chờ ban đầu (chưa chạy game)
- Xử lý:
    - Hiển thị màn hình chờ
    - Khi nhấn phím bất kỳ, bắt đầu tạo đối tượng, khởi động game loop
- Đẩu ra: chương trình bắt đầu chạy

2.3. Cấu trúc dữ liệu

- Class:

Class	Mô tả
Ship	Đại diện cho tàu người chơi. Quản lý góc quay, tốc độ, hình ảnh, vị trí và bắn tên lửa.
Missile	Đại diện cho tên lửa được bắn ra. Di chuyển theo hướng của tàu, kiểm tra va chạm.

Class	Mô tả
Asteroid	Đại diện cho tiểu hành tinh. Tự di chuyển theo hướng và tốc độ ngẫu nhiên.
Explosion	Hiệu ứng nổ xuất hiện khi có va chạm. Tồn tại trong thời gian ngắn.

- Sprite:

Biến	Kiểu dữ liệu	Vai trò
all_sprites	pygame.sprite.Group()	Chứa tất cả các sprite, dùng để cập nhật và vẽ.
missiles	pygame.sprite.Group()	Chứa các tên lửa, dùng để kiểm tra va chạm với asteroid.
asteroids	pygame.sprite.Group()	Chứa các asteroid. Tạo, cập nhật, kiểm tra va chạm.
explosions	pygame.sprite.Group()	Chứa các hiệu ứng nổ tạm thời.

- Biến cục bộ:

Biến	Kiểu dữ liệu	Vai trò
score	int	Điểm của người chơi. Tăng khi bắn trúng asteroid.
running	bool	Kiểm soát vòng lặp game chính.
clock	pygame.time.Clock()	Điều chỉnh tốc độ khung hình.
screen	pygame.Surface	Màn hình hiển thị chính.

- Tài nguyên (Resource Data):

Loại	Dữ liệu	Lưu trong biến
Ảnh	PNG	ship.png, missile.png, asteroid.png, background.png, explosion.png
Âm thanh	WAV	shoot.wav, boom.wav
Nhạc nền	MP3	background_music.mp3

- Dữ liệu vị trí, góc, chuyển động:

Thành phần	Biến quan trọng
Vị trí ship	rect.x, rect.y, rect.center
Góc quay	angle (độ), dùng trong xoay sprite và hướng chuyển động
Vận tốc	speed, cos(angle), sin(angle) tính toán hướng bay

Thành phần	Biến quan trọng
Vị trí đầu tàu	Tính bằng offset từ rect.center dùng trong get_head_position()

- Game states:

- waiting_for_key: cờ kiểm soát việc chờ người chơi nhấn phím để bắt đầu game.
- game_over: trạng thái kết thúc, hiển thị màn hình "Game Over".
- pygame.KEYDOWN, pygame.QUIT: sự kiện điều khiển game loop.

2.4. Các hàm trong chương trình

- Các hàm load_image(filename, size=None), load_sound(filename), load_music(filename): tải các file hình ảnh, âm thành và nhạc

- Hàm trong class ship:
  - __init__(self): Khởi tạo tàu
  - update(self): Xử lý dữ liệu từ phím và cập nhật vị trí, hướng tàu
  - get_head_position(self): Xác định vị trí đầu tàu
  - shoot(self): Tạo missile từ đầu tàu

- Hàm trong class Missile:
  - __init__(self, x, y, angle): Khởi tạo tên lửa tại vị trí và hướng cụ thể
  - update(self): Cập nhật vị trí missile

- Hàm trong class Asteroid:
  - __init__(self): Tạo asteroid với vị trí và hướng ngẫu nhiên
  - update(self): Di chuyển asteroid theo hướng đã định

- Hàm trong class Explosion:
  - __init__(self, x, y): Khởi tạo hiệu ứng nổ tại tọa độ
  - update(self): Giảm thời gian tồn tại của đối tượng Explosion

CHƯƠNG 3: THỰC NGHIỆM VÀ KẾT LUẬN

3.1. Thực nghiệm

Sau khi hoàn thành chương trình, tiến hành chạy thử, thực nghiệm thu được kết quả của các chức năng:

- điều khiển tàu
- bắn tên lửa
- tính điểm
- tạo asteroid
- dừng chương trình

3.1.1. Điều khiển tàu

Phím mũi tên lên tăng tốc độ tàu, phím mũi tên xuống giảm tốc độ tàu, phím mũi tên trái phải điều chỉnh hướng tàu

- Tăng tốc độ tàu, tiến lên:

Hình 1: tàu ở vị trí ban đầu

Hình 2: tàu tăng tốc tiến lên

Kết quả: chương trình chạy đúng yêu cầu

- Giảm tốc độ tàu, lùi lại:

Hình 3: Tàu ở vị trí ban đầu

Hình 4: tàu giảm tốc, lùi lại

Kết quả: chương trình chạy đúng yêu cầu

- Điều chỉnh hướng tàu:

Hình 5: tàu ở vị trí ban đầu

Quay trái:

Hình 6: tàu quay sang trái

Quay phải:

Hình 7: tàu quay sang phải

Kết quả: chương trình chạy đúng yêu cầu

3.1.2. Bắn tên lửa

Bấm phím cách thì tên lửa được bắn ra từ mũi tàu, khi tên lửa chạm asteroid thì asteroid bị phá hủy

Hình 8: tàu ở vị trí ban đầu

Hình 9: tên lửa bắn ra từ tàu theo hướng mũi tàu

Hình 10: tên lửa trúng asteroid

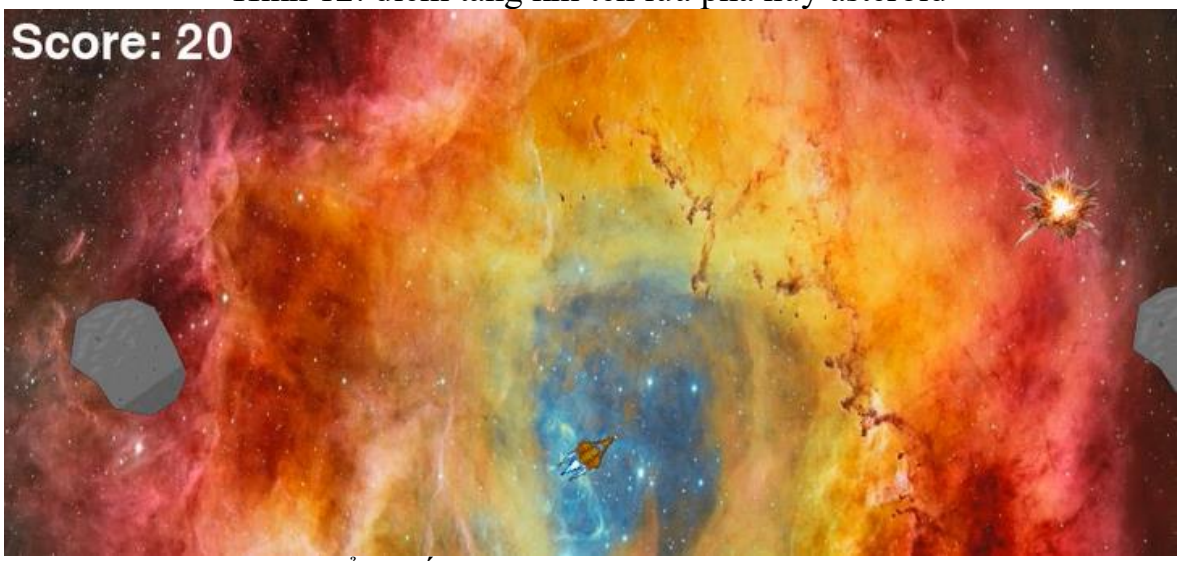
Kết quả: chương trình chạy đúng yêu cầu

3.1.3. Tính điểm

Điểm khi bắt đầu chương trình là 0, khi tên lửa phá hủy 1 asteroid thì điểm tăng thêm 10

Hình 11: Điểm khi bắt đầu chương trình

Hình 12: điểm tăng khi tên lửa phá hủy asteroid

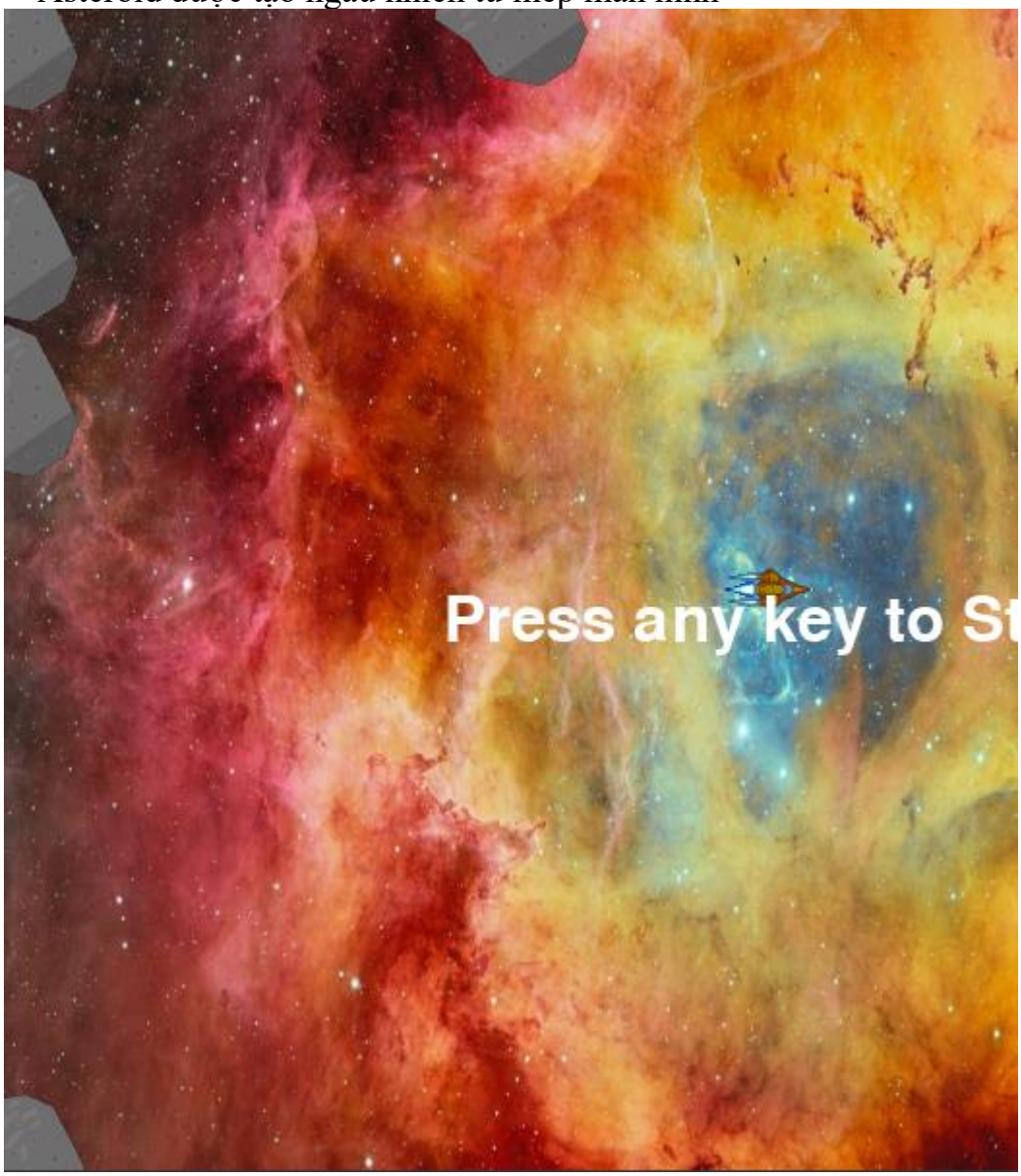


Hình 13: điểm tiếp tục tăng khi tên lửa phá hủy asteroid

Kết quả: chương trình chạy đúng yêu cầu

3.1.4. Tạo asteroid

Asteroid được tạo ngẫu nhiên từ mép màn hình

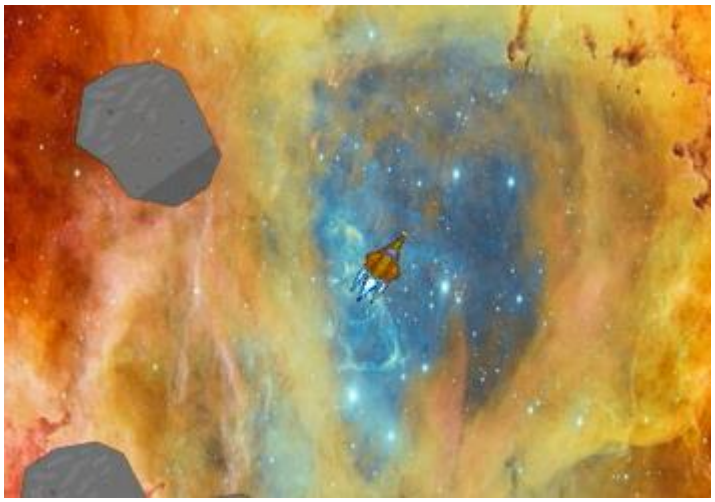


Hình 14: Asteroid được tạo ngẫu nhiên từ mép màn hình

Kết quả: chương trình chạy đúng yêu cầu

3.1.5. Dừng chương trình

Chương trình dừng khi tàu và asteroid va chạm



Hình 15: chương trình đang chạy

Hình 16: chương trình dừng khi tàu và asteroid va chạm

Kết quả: chương trình chạy đúng yêu cầu

3.2. Kết luận

3.2.1. Những gì sản phẩm làm được.

Chương trình hoạt động đúng yêu cầu.

Chương trình hoạt động ổn định.

3.2.2. Những gì đã học được.

Xây dựng một chương trình gồm nhiều thành phần phương tiện như âm thanh, âm nhạc, hoạt ảnh.

Sử dụng các tài nguyên hiệu ứng âm thanh, âm nhạc trong chương trình.

Sử dụng class Sprite giúp quản lý các đối tượng hình ảnh, cập nhật trạng thái của đối tượng và hiển thị lên màn hình.