BÁO CÁO

LẬP TRÌNH GAME BẮN TRỨNG KHỦNG LONG

Mục lục

[Chương I: Giới thiệu chung về đề tài 1](#_Toc93429347)

[Chương II: Chi tiết của sản phẩm bài làm 2](#_Toc93429348)

[1. Cơ sở của lập trình 2](#_Toc93429349)

[1.1. Thư viện và các hàm 2](#_Toc93429350)

[1.2. Thuật toán 3](#_Toc93429351)

[2. Các thành phần của chương trình 6](#_Toc93429352)

[2.1. Khái quát chung các thành phần 6](#_Toc93429353)

[2.2. Các lớp và công dụng 7](#_Toc93429354)

[2.2.1. Arrow 7](#_Toc93429355)

[2.2.2. Egg 7](#_Toc93429356)

[2.2.3. Score 7](#_Toc93429357)

[2.2.4. Settings 7](#_Toc93429358)

[2.2.5. Menu 7](#_Toc93429359)

[2.2.6. Dino 7](#_Toc93429360)

[2.2.7. Engine 7](#_Toc93429361)

[Chương III: Kết quả 7](#_Toc93429362)

# Chương I: Giới thiệu chung về đề tài



Bắn trứng khủng long – Dynomite delux là một trò chơi trên PC được phát triển bởi Raptisoft Games và được phát hành bởi PopCap Games. Trong game này, trứng gồm nhiều màu sắp được xếp thành bảng rơi dần từ trên xuống, nhiệm vụ của bạn là bắn vỡ những quả trứng bằng cách tạo thành tối thiểu 3 quả có màu sắc giống nhau, khi trứng vỡ bạn sẽ được cộng điểm, bắn vỡ càng nhiều điểm càng cao. Trò chơi kết thúc khi bạn để bảng trứng chạm vào vạch kẻ phía dưới

# Chương II: Chi tiết của sản phẩm bài làm

1. Cơ sở của lập trình
   1. Thư viện và các hàm

Phần lớn game này đã chạy được nhờ thư viện “pygame”.Ngoài ra sử dụng các hàm trong math như các hàm tính góc sin, cos, tan, sử dụng copy, deepcopy trong thư viện copy để copy các list

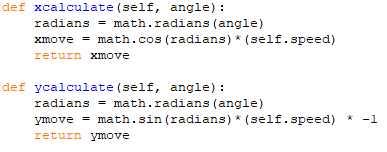
Pygame hỗ trợ nhiều chức năng hữu dụng như tính toán thời gian, fps, hỗ trợ các hàm để in đồ họa ra màn hình(blit()), hàm kiểm tra va chạm trong lớp Sprites(collide\_rect())....

Nếu chỉ có hình ảnh không thôi mà không có âm thanh thì trải nghiệm chơi game sẽ trở nên nhàm chán đi gấp nhiều lần. Nhờ có pygame, ta cũng có thể chạy được những tiếng động bằng câu lệnh pygame.mixer.Sound(). Lệnh này sẽ tạo một đối tượng Sound, âm thanh từ 1 file tiếng (ogg, mp3...) từ đó có thể được dùng để gọi trong code, ở đây sử dụng file(.ogg) để giảm dung lượng game.

Ví dụ: Tiếng trứng vỡ



Thư viện math cung cấp các hàm tính toán để tính góc bắn, tọa độ của trứng



Module random được dùng cho việc đảo ngẫu nhiên các màu trứng

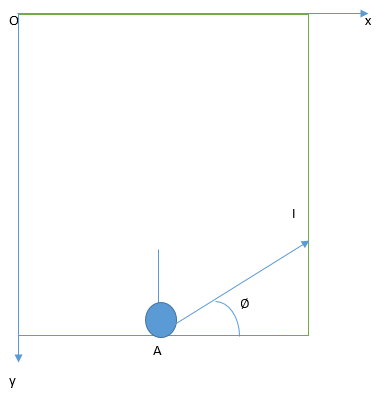


* 1. Thuật toán
* Vấn đề 1:

Để biết được quả trứng được bắn ra sẽ đi đâu, đi theo hướng nào thì ta cần tính được góc bắn của quả trứng.

Quả trứng sẽ được bắn lên từ dưới lên, sẽ có 2 trường hợp xảy ra:

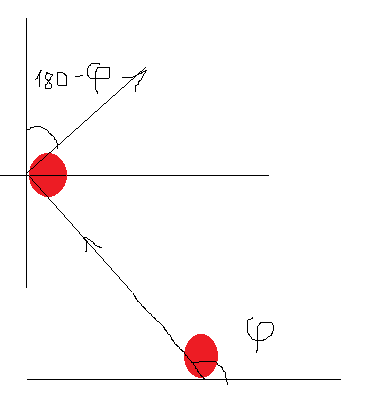
* Trường hợp 1: bắn góc 90 độ thì ta chỉ cần giảm dần tọa độ y, tọa độ x giữ nguyên
* Trường hợp 2: góc bắn < 90 độ:
* x =
* y = (vì trục Oy chiều từ trên xuống dưới nên nhân thêm -1)
* Trường hợp 3: góc băn > 90 độ:
* x = (khi góc lớn hơn 90 độ bóng sẽ đi ngược chiều Ox)
* y = y =



*Hình 1. Minh họa góc bắn*

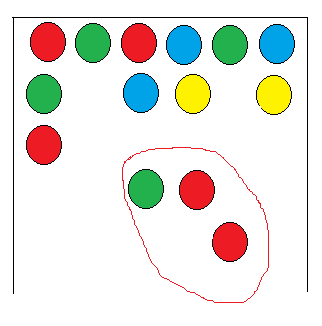
Khi trứng chạm tường ta tính lại góc bay của trứng lúc này sẽ bằng

trong đó là góc bắn ban đầu(



*Hình 2.Góc phản xạ khi chạm tường*

* Vấn đề 2: Khi ta bắn trứng những mảng trứng nổi không dính vào phần mảng trứng lớn sẽ cũng bị loại bỏ

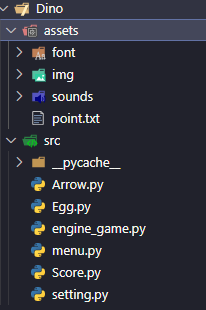


*Hình 3. Trứng nổi cần xóa*

Để xóa được trứng nổi ta cần sử dụng thuật toán Flood fill.

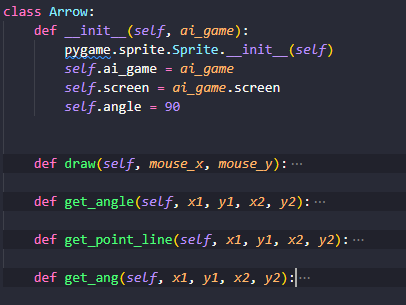
Flood fill là một thuật toán nhằm xác định các thành phần kết nối với nhau trong đồ thị hoặc trong một khu vực bất kì. Ta sẽ chạy thuật toán từ hàng đầu tiên của bảng trứng kiểm tra tất cả các quả trứng liên thông với nhau thuộc mảng lớn này, những quả khác không liên thông với mảng lớn sẽ là những quả bị xóa bỏ

1. Các thành phần của chương trình
   1. Khái quát chung các thành phần



*Hình 4.Các thành phần của chương trình*

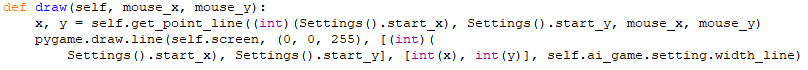
* Folder game gồm 2 folder lớn đó là assets và src.
* Assets lưu trữ các tài nguyên game như ảnh, font chữ, âm thanh,…
* Src lưu trữ các class trong game.
* Các class trong game gồm:
* Arrow: đường thẳng định hướng bắn trứng
* Egg: những quả trứng
* Score: lớp này để tính điểm
* Menu: lớp menu của game
* Dino: lớp game
* Setting: lớp lưu các thiết lập, thông số game
* Engine: lớp điều khiển game
  1. Các lớp và công dụng
     1. Arrow



*Hình 5. Lớp Arrow*

Lớp này dùng để vẽ đường định hướng bắn của trứng, có 4 hàm chính là draw, get\_angle, get\_point\_line, get\_angl.

* **draw():**



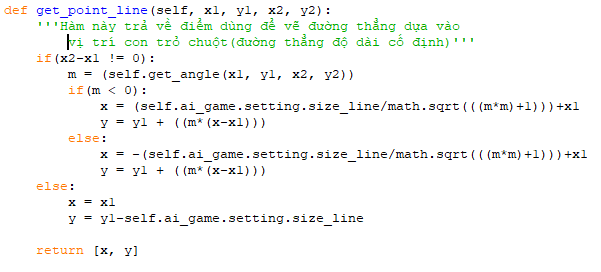
Dùng để vẽ đường thẳng độ dài cố định nối từ quả trứng đạn tới vị trí tỉ lệ với vị trí chuột

* **get\_angle()**:



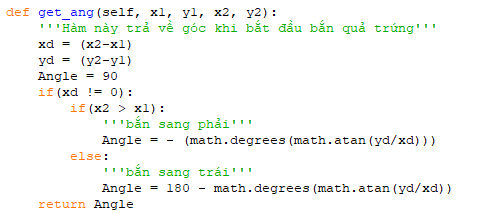
* **get\_point\_line:**

Hàm này trả về vị trí đầu mũi tên của đường thẳng tỉ lệ với vị trí chuột



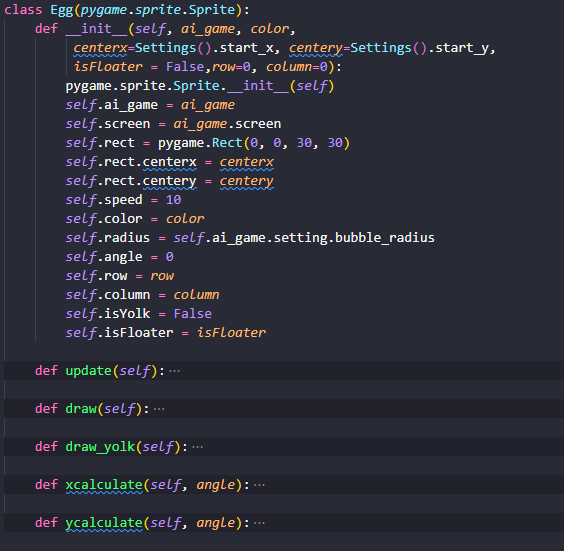
* **get\_ang()**:

Hàm này trả về góc bắn trứng theo độ



* + 1. Egg

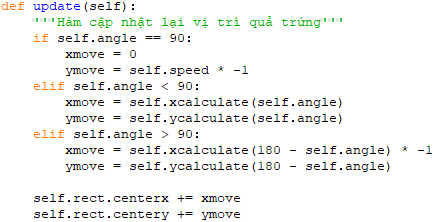
Lớp này tạo ra các quả trứng sử dụng trong game



*Hình 6. Lớp Egg*

* **update():**

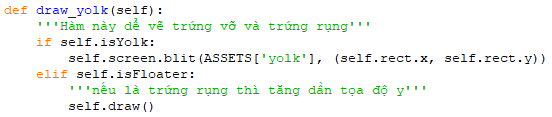
Hàm này cập nhật lại vị trí của trứng khi bắn



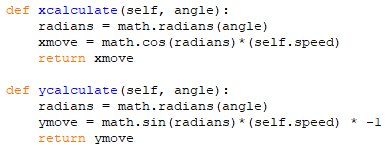
* **draw():** vẽ trứng dựa vào màu sắc



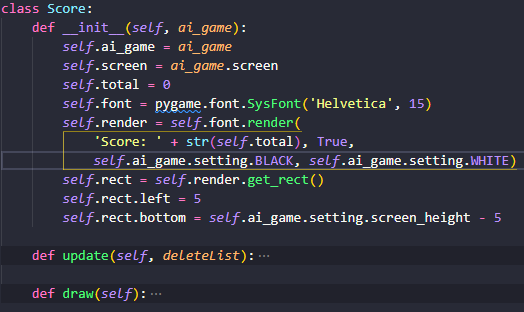
* **draw\_york():** vẽ những quả trứng vỡ và quả trứng rơi



* Hàm tính tọa độ x, y của trứng dựa vào góc:

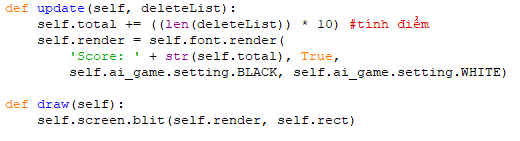


* + 1. Score

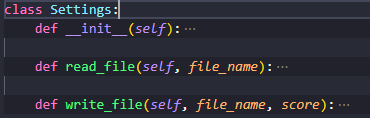


*Hình 7. Lớp Score*

Lớp này tính điểm dựa vào số trứng bắn được, cập nhật lại điểm bằng hàm **update()** và in điểm ra màn hình bằng hàm **draw()**

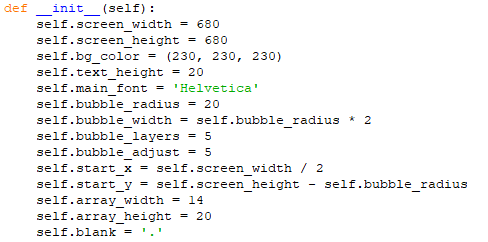
****

* + 1. Settings

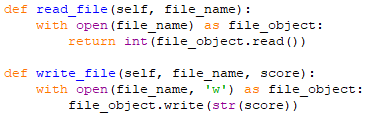


*Hình 8 Lớp Settings*

* Lớp này lưu các thông số của game như kích thước màn hình, kích thước trứng, ….



* Lớp này còn có thêm 2 hàm đọc và ghi file để ghi điểm vào file



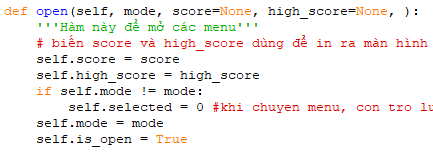
* + 1. Menu

Menu là lớp tạo ra menu game, trong lớp này chứa các menu option là các chức năng của game, người chơi có thể chọn bắt đầu chơi hoặc thoát khỏi game, khi game đang chơi, người chơi có thể pause game, hoặc chuyển sang menu bắt đầu trò chơi mới



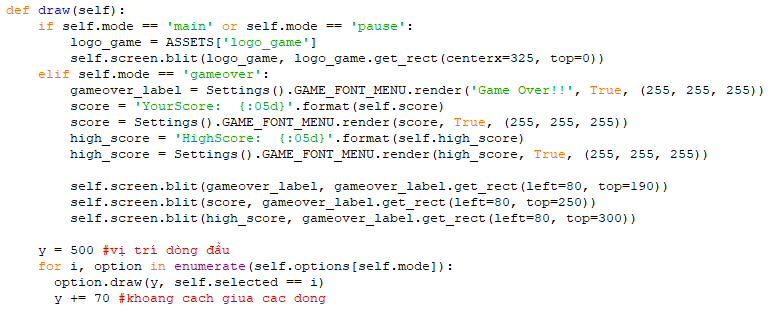
*Hình 9. Lớp Menu*

* **open():**



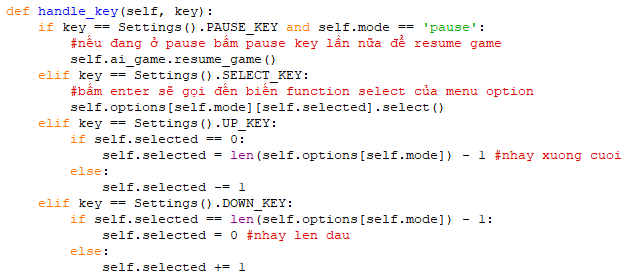
Hàm này có chức năng mở các option bằng cách sử dụng biến selected, selected tương ứng với mỗi option khác nhau, option đầu tiên có selected=0

* **draw():**



In ra các option lên màn hình và in ra màn hình gameover

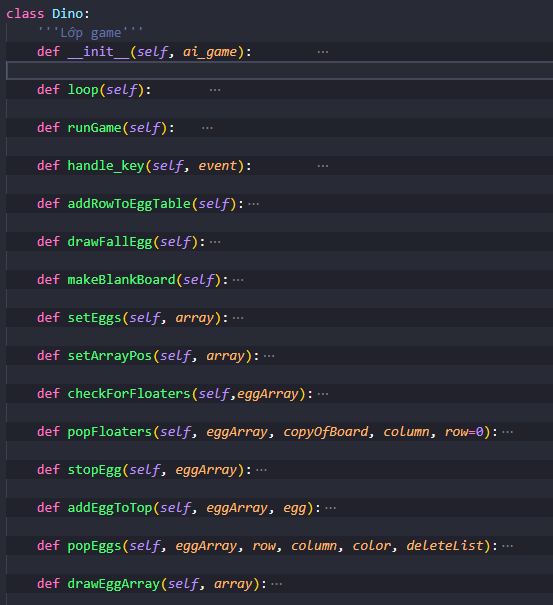
* **handle\_key():**



Hàm này xử lý nút bấm khi người chơi bấm nút mũi tên lên xuống, con trỏ vào option cũng thay đổi theo

* + 1. Dino

Đây là lớp xử lý chính của game, lớp này khởi tạo trò chơi và điều khiển các hoạt động chơi như tạo bảng trứng, bắn trứng,…

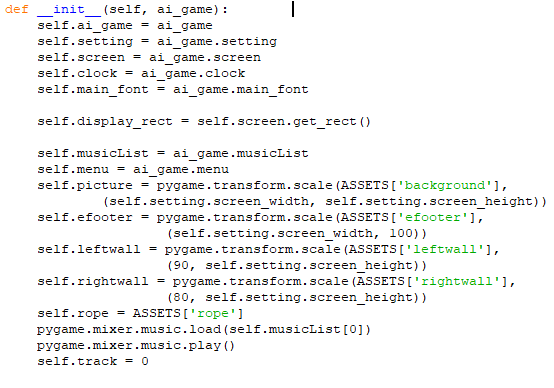


*Hình 10. Lớp Dino*

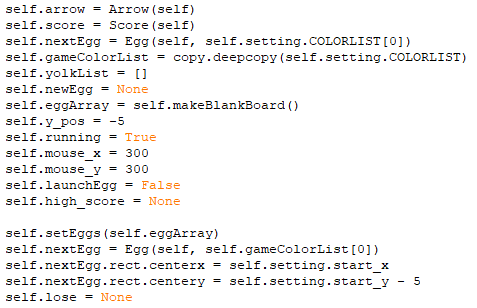
* Hàm khởi tạo:

Truyền vào hàm một lớp là lớp engine game, trong đó gồm các thuộc tính như màn hình hiển thị, fps,..

Tiếp theo khởi tạo tường, background, và nhạc nền,..

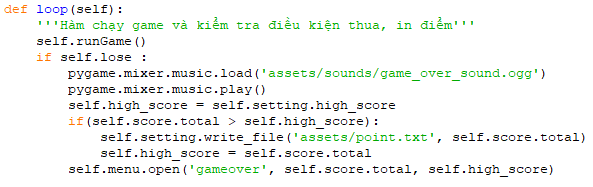


Sau đó, khởi tạo tiếp các đối tượng cần thiết của game đó là bảng trứng, đạn trứng, mũi tên định hướng,…



* **loop():**

Hàm này gọi đến hàm chạy game, và khi game over, nó sẽ cập nhật lại điểm cao nhất trong file và chuyển qua màn hình menu game over

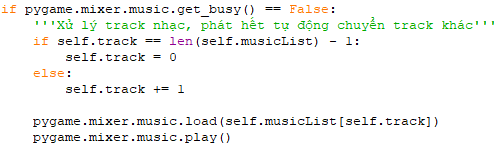


* **RunGame():**

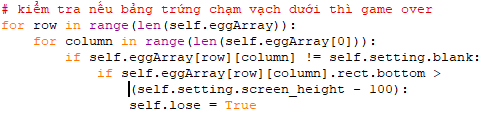
Hàm này làm nhiệm vụ xử lý game, như xử lý bắn trứng, xử lý hình ảnh âm thanh, bắt các sự kiện nhấp chuột, … bằng cách gọi đến các hàm chức năng khác được nêu bên dưới.

* Các ví dụ về chức năng xử lý trong hàm run game:
* Xử lý track nhạc

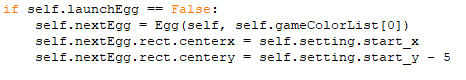
Nếu track phát xong thì nó sẽ tự động chuyển sang track tiếp theo trong list các track



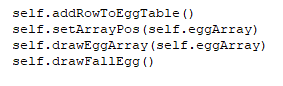
* Kiểm tra điều kiện thua:



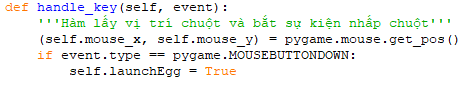
* Tạo ra quả trứng mới ở vị trí trung tâm phía dưới màn hình khi bắn xong quả cũ



* Gọi đến các hàm xử lý khác



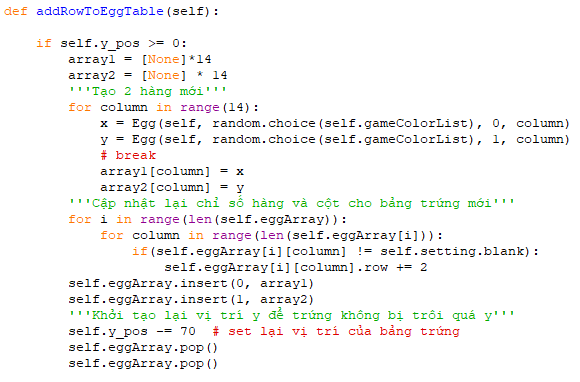
* **handle\_key():**



Hàm này có nhiệm vụ bắt sự kiện nhấp chuột và nhận vị trí của con trỏ chuột hiện tại, nếu nhấp chuột sẽ bắn trứng.

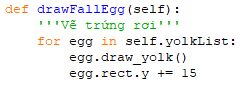
* **addRowToEggTable:**

Khi bảng trứng rơi xuống vị trí y nhất định, hàm này sẽ thêm 2 hàng mới vào đầu bảng trứng, tạo hiệu ứng trứng rơi dần từ trên xuống, và lấp kín khoảng trống phía trên, thêm 2 hàng bởi vì bảng trứng được xếp sole, nếu thêm lẻ hàng sẽ làm vị trí của trứng cũ thay đổi. Đồng thời, khi thêm hàng mới vào, ta cũng phải cập nhật lại vị trí hàng và cột của những quả trứng cũ trong bảng trứng



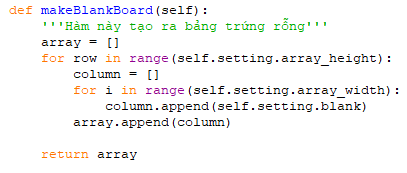
* **drawFallEgg():**

Hàm này thực hiện chức năng vẽ những quả trứng bị bắn rơi

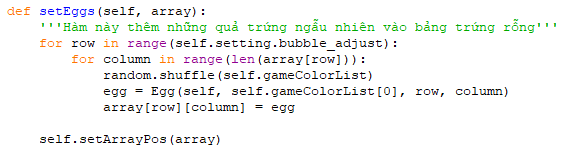


* **makeBlankBoard():**

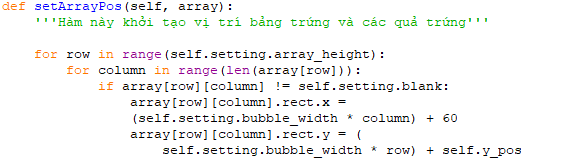
Hàm trả về bảng rỗng

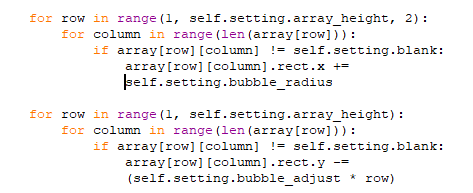


* **setEggs(), setArrayPos():**



Hàm **setEgg()** có chức năng tạo ra các quả trứng mới ngẫu nhiên và thêm vào bảng trứng, sau đó nó gọi đến hàm **setArrayPos()** để khởi tạo vị trí của bảng trứng

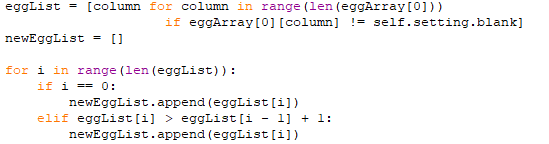




* **checkForFloaters()**

Hàm này làm nhiệm vụ kiểm tra những quả trứng nổi không liên thông với mảng trứng lớn, và thêm chúng vào list trứng bị xóa.

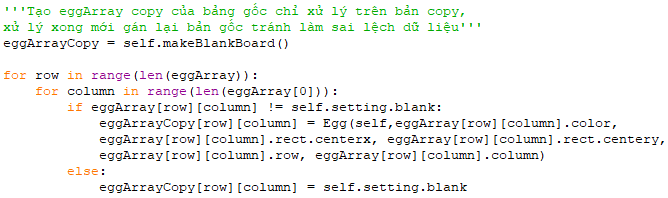
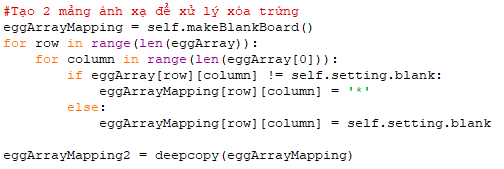
Đầu tiên, cần tìm xem những cột cần duyệt, những cột này là những cột khác rỗng



Sau đó tạo ra 2 ma trận ánh xạ và 1 ma trận copy tương ứng với ma trận trứng ban đầu để xử lý xóa trên ma trận ánh xạ và ma trận copy này sau đó mới cập nhật trên ma trận gốc

Trong đó một ma trận ánh xạ chỉ chứa toàn kí tự rỗng để thực hiện thuật toán Flood fill trên ma trận này.

(Cần phải xử lý gián tiếp trên ma trận copy và ma trận ánh xạ vì deepcopy không thể tạo ra bản copy của các class trong pygame)

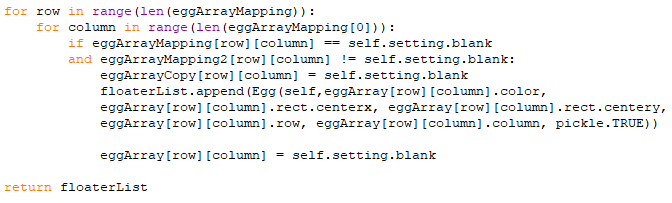
Gán toàn bộ các phần tử bằng phần tử rỗng để thực hiện Flood fill



Ta thực hiện Flood fill trên những cột cần thiết:

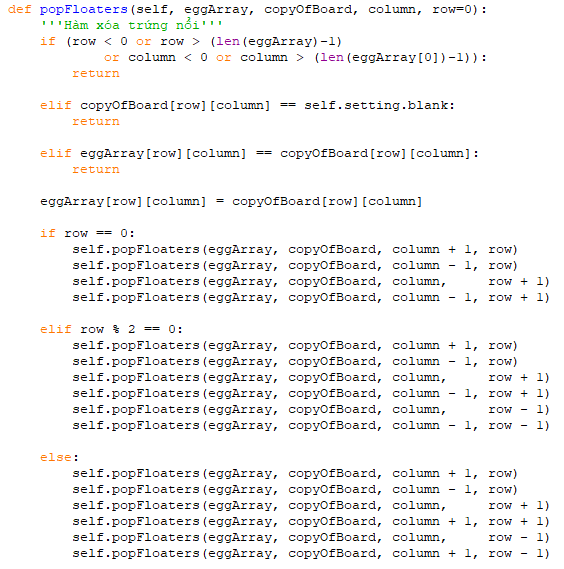


Sau khi thực hiện xong thu được ma trận ánh xạ mới đã bỏ đi những phần tử là trứng nổi, tiếp theo cập nhật lại bảng ma trận trứng ban đầu và thêm những quả trứng bị bỏ đi vào floaterList để in chúng ra màn hình tạo hiệu ứng rơi



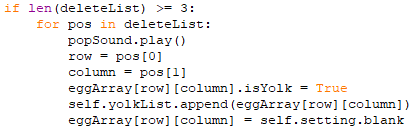
* **popFloaters():**

Hàm này sử dụng thuật toán Flood fill để điền dần vào bảng rỗng những ô có thể điền được, đó là những ô liên thông với bảng trứng lớn, còn những mảng nhỏ không liên thông với bảng lớn sẽ bị bỏ qua

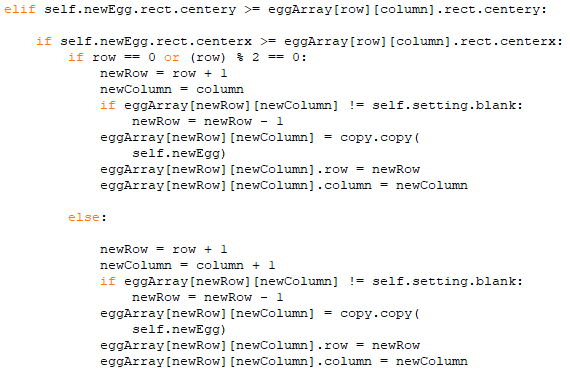


* **stopEgg():**

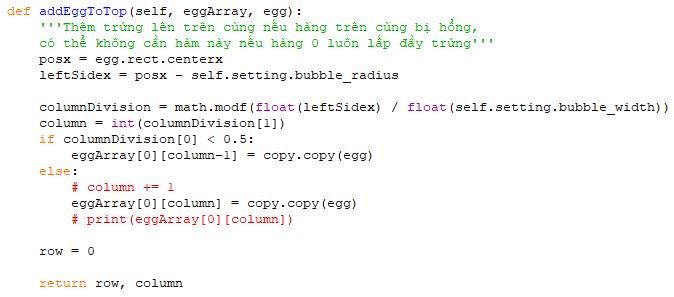
Hàm này có chức năng kiểm tra xem trứng bắn ra đã chạm vào bảng trứng hay chưa, nếu rồi thì thêm quả trứng mới vào bảng, nếu trứng chạm phải trứng cùng màu và tạo ra 3 quả cùng màu trở lên thì trứng sẽ vỡ.



Nếu trứng không vỡ thì sẽ thêm trứng vào vị trí trống, nếu lệch dưới thì thêm bên dưới, lệch phải thêm bên phải,….

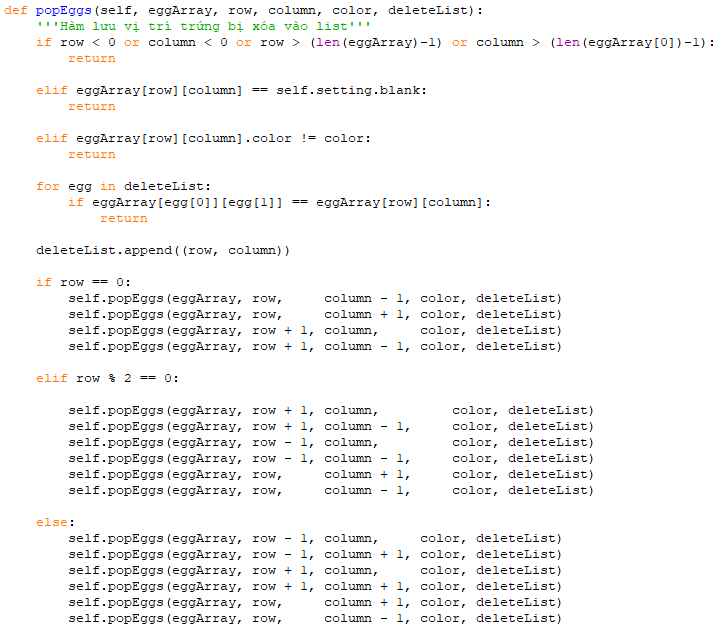


Trong trường hợp trứng bắn lên trên đầu màn hình và hàng 0 rỗng thì gọi hàm **addEggToTop()**

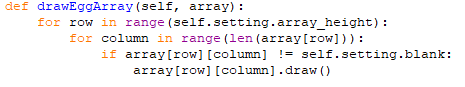


* **popEggs():**

Hàm này có nhiệm vụ xóa đi những quả trứng cùng màu bằng thay vị trí những quả trứng cần xóa bằng rỗng

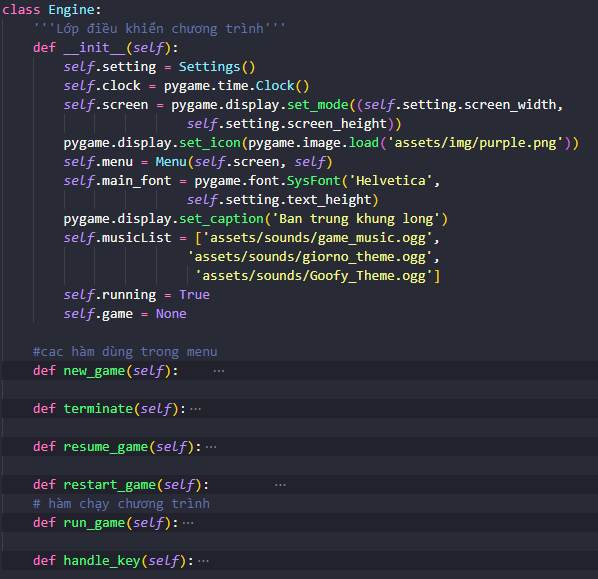


* **drawEggArray():** hàm vẽ bảng trứng ra màn hình

****

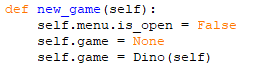
* + 1. Engine

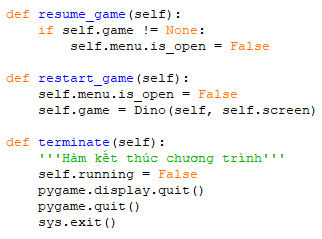
Lớp này là lớp engine game điều khiển hoạt động của game, menu



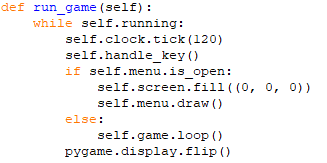
*Hình 11. Lớp Engine*

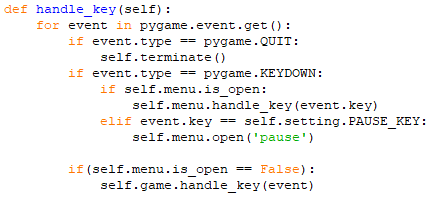
* Hàm khởi tạo: khởi tạo lớp Setting, clock, và menu nhạc game, font chữ cùng các thông số chung như kích thước màn hình
* Các hàm dùng trong menu: gồm các hàm tạo mới game, bắt đầu game mới, chơi tiếp





* Hàm **run\_game()** dùng để khởi tạo fps và chạy chương trình, nó gọi đến hàm handle\_key để bắt các sự kiện khi người chơi bấm các nút và chuột

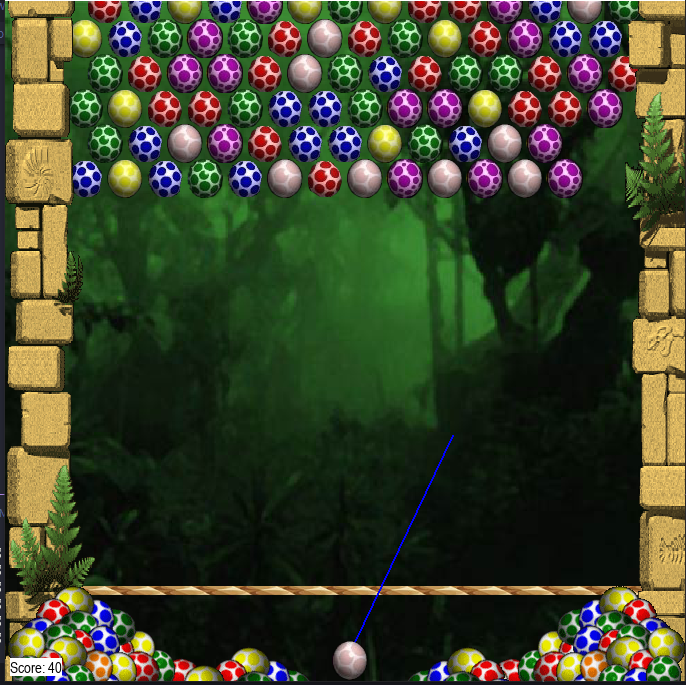




# Chương III: Kết quả



*Hinh 12.Giao diện menu khi bắt đầu game*

*Hình 13.* *Giao diện khi chơi game*



*Hình 14. Giao diện menu khi pause game*



*Hình 15. Giao diện game over*