Pseudo code

Alien

Function : start (creation of the aliens)

Time Complexity: O(N), where N = cte.NB_MAX_ALIEN. It iterates through the range N to create aliens.

Space Complexity: O(N) for the list fleet.

Function : get_alien_contact

Time Complexity: O(N), where N is the number of aliens in the fleet. It iterates through each alie in the fleet.

Space Complexity: O(1).

Function : move

Time Complexity: O(1) per invocation, but since it recursively calls itself using ontimer, the total runtime depends on game logic (the number of moves before aliens exit the screen or the game ends).

Space Complexity: O(1) per invocation.

Bomb

Function : move

Time Complexity: O(1) per invocation. However, since it is a recursive function because of ontimer, the total time complexity depends on the number of moves required before the bomb goes off-screen or collides.

Space Complexity: O(1)

Function :

check_collision

Time Complexity: O(1), assuming the get_ship_contact function is constant time.

Space Complexity: O(1).

Bullet

Function : move

Time Complexity: O(1) per invocation. However, the total runtime depends on the number of moves the bullet makes before exiting the screen or colliding with an alien.

Space Complexity: O(1) per invocation, as no additional memory is needed.

Function : check_collision

Time Complexity: O(N), where N is the number of aliens in the fleet. This function calls get_alien_contact, which iterates over the fleet.

Space Complexity: O(1).

Game

Function : new_game

Time Complexity: O(N), where N is the number of aliens, as it removes the alien fleet and initializes a new one.

Space Complexity: O(N), for the newly created alien fleet.

Function : next_level

Time Complexity: O(N), where N is the number of aliens, as it clears and initializes a new alien fleet.

Space Complexity: O(N), for the newly created alien fleet.

Function :end_game

Time Complexity: O(M+T), where M is the number of aliens to remove, and T is the number of top scores to display.

Space Complexity: O(T), for storing the top scores.

Function : get_top_scores

Time Complexity: O(K+Tlog(T)), where K is the number of scores in the file, and T is the number of top scores to extract.

Space Complexity: O(K), for reading and storing all scores.

Ship

Function :move

Time Complexity: O(1), as it updates the position of the ship based on the input key and performs simple boundary checks.

Space Complexity: O(1).

Function :fire

Time Complexity: O(1) per invocation. However, creating a Bullet and invoking its move method will trigger recursive behavior in the Bullet class (analyzed separately).

Space Complexity: O(1) per invocation, except for appending to the bullet_list, which grows by O(1) per bullet.

Space_invaders

Function : update

Time Complexity: O(1) per invocation. The function updates the screen and schedules the next update via ontimer. However, its cumulative complexity depends on how long the game runs.

Space Complexity: O(1) per invocation, as it doesn't allocate additional memory.

Function : space_invaders

Time Complexity: O(N), where N is the number of aliens created by the Alien_fleet. The initialization of the game and objects like Ship and Alien_fleet dominate this function’s complexity.

Space Complexity: O(N), proportional to the number of aliens in the fleet.