**"SPACE INVADERS"**

**By**

**Ummar Ikram**

**World:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | aaa |  | sss |  | ccc |  | ooo |  | eee |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | . |  |  |  |  |  |
|  | zzz |  | mm |  | nnn |  | uuu |  | rrr |
|  |  |  |  |  |  |  |  |  |  |
|  | . |  |  | ^ |  |  | . |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | === |  |  |  |  |  |

SCORE: LIVES:

A 2-D [25][80] Array of Characters with the following:

‘=’: Character used to represent Player in the 2-D Array.

‘a/s/c/o/e/z/m/n/u/r’: Characters used to represent Enemy in the 2-D Array.

‘ ’: Character used to represent empty space in the 2-D Array.

‘ ^ ’: Character used to represent Player Laser in the 2-D Array.

‘ . ’: Character used to represent Enemy Laser in the 2-D Array.

**Player Movement:**

The user inputs a key and depending on the input a series of action can take place.

‘a’ or ‘A’: If either of these are pressed the Player represented by ‘=’ moves 1 Block Leftwards.

‘d’ or ‘D’: If either of these are pressed the Player represented by ‘=’ moves 1 Block Rightwards.

‘Space’: If Space is pressed the Player represented by ‘=’ shoots a Laser represented by ‘^’ 1 Block Upwards from its current position.

**Enemy Movement:**

**A random moving time for moving enemies, both vertically and horizontally. If any enemy is near the left border, all of them fall vertically rightwards \. If any enemy is near the right border, all of them fall vertically leftwards /.**

**Else they continue to move horizontally in either right or left direction.**

**This continues until they are parallel to player’s position that is when they have invaded successfully.**

**Enemy Shooting:**

A random time is chosen using Random Function for the Enemy to Shoot 1 Block Downwards from its Position. This removes multiple shooting by enemies and a random enemy shoots each time for a more natural feeling.

An Enemy does not shoot if another Enemy is directly below him (Controlled by Enemy Ready variable). Once the Enemy below is killed, the Enemy above is free to shoot.

If an Enemy Laser ‘.’ is intercepted by a Player Laser ‘^’ both Lasers cancel out each other.

If an Enemy Laser ‘.’ hits Player ‘=’ , Enemy Laser ‘.’ Vanishes and Player Lives ‘ ’ Decrease.

Else Enemy Laser ‘.’ continues Downwards.

**Player Shooting:**

If the Player Laser ‘^’ hits an Enemy Laser ‘.’ Both Lasers cancel out each other.

If the Player Laser ‘^’ hits an Enemy, Enemy is killed and Player Laser ‘^’ disappears. Player Score is increased by 20 and Enemy Count is decreased.

Player Laser ‘^’ is also controlled by Laser Ready variable to avoid multiple shooting by Player ‘=’.

Else Player Laser ‘^’ continues Upwards.

**Shapes:**

Player ‘=’ is covered with Green Color Rectangle that moves along with Player ‘=’ and is cleaned when Player Lives are 0.

Player Lives are represented by Green Color Rectangle(s) and are cleaned chronologically when Player Lives Decrease.

Borders are covered with White/Cream Color Lines and are cleaned when Game Ends.

**Game End:**

Game ends when all enemies are killed or no player lives left or Enemies are parallel to player position.

An Output of Total Score is displayed after game ends.

**Further Improvements:**

* Barriers can be added that prevent both Enemy Laser & Player Laser from Passing through.
* High Health & Score Enemies can be added.
* Score saving system/Leaderboards can be added to a separate file.

**Code Performance:**

Meaningful variable/function names are used.

Entire code is well commented. // Comment.

Easy logic & shorter code with separate functions for every mechanism.

Use of 2 nested for loops & a number of selection statements.

**Time Complexity:**

Not Calculated. But because of independent nested for loops it’s around - O (n^2).

**THANK YOU!**