<u>Snek</u>

As the title suggests, you are a snake who eats apples while avoiding the garden walls and his own body.
Programmed in python with the help of some tools.

<u>Tools used</u>:
1) **Pygame** : A python game developing library based on sdl2.
2) **Libresprite** : An open-source fork of asperite, a software used to develop high quality pixel art.
3) **SFXR** : Sound effects maker and generator. Frequently used in game jams.
4) **GitHub**: A host that uses git's software development and version control systems to store repositories.

<u>Source Files</u>:
This game is divided into 3 source files for now.
1) The main game file: This is where the game works.
2) The menu file: When the menu is displayed, it gives the player three options as of now. Upon clicking, first one starts a new game, second one takes you to the level editor, and the third one exits the game.
3) The editor file: With this editor, you can create and edit your custom levels using the mouse, left click to place the wall tile and right click to remove it from the selected position.

<u>Working of the main game</u>:
This game uses a list to store the snake's body, where each element has a image,a position and a direction. The direction is to make the tail spawn correctly after eating an apple. Then the tile attributes are created following the map, column-wise(A column wise interpretation of the map will not look intuitive, but working with it is, especially for those who are good at math)

The game uses a short time frame of 6 frames/movement to make the game playable, all movement occurs in the correct key only. Furthermore to keep up the polish, the movements occur in a map, which is divided into 32*32 segments. Any movement will take the snake to the next segment instead of the pixel.

Once the game loop starts, the snake starts at a fixed coordinate facing right. Then the apple to be eaten is randomly created in a position such that it does not spawn inside the snake or the wall. Then the arrow keys steer the snake by changing the direction of its head. Then, after the correct time frame, the movements are applied.

Since the game uses 32*32 segments and 32*32 images as its data, collision is only checked after the correct time frame and the check is one of the simplest ones. When the snake makes the move, collision is checked for whether the tile the snake is about to hit contains nothing(0),air(1), a piece of its own body(2) or a wall(3)
If value is 0, move ahead.
If value is 1, snake eats the apple, then a new apple spawns, then the snake grows in length
If value is 2 or 3, its game over.

For the purpose of speed, some adjustments are done
1)whenever the snake eats an apple, instead of transferring the properties of the previous segments one by one,
it instead inserts a new tail with the attributes of the old tail and then converts the old tail into a body segment
2) whenever the snake moves, the tile attribute of the tail is set to 0, then the old tail is removed, then the last segment of the snake’s body is set to a tail,retaining its attributes, then a new head segment is added to where the snake should move with the proper attributes, then the tile attribute of the head is set to 2 for collision checks to

work properly, instead of transferring the attributes segment by segment.

Finally after all this, the state of the game is rendered as such, and then the game loop continues till the user quits or gets a game over.

Working of the editor:
Simple. A left click places the wall tile in the chosen segment, a right one removes it. Once the editing is done, the user can press <Esc> to pause the editor, following which he can click on y to save the level, n to exit without saving, or c to continue work. When the user chooses to save the level, he is greeted with a small dialog to give the level a name, which will be saved in the custom directory as “name.map”. Extension checks are done accordingly to recognize the name. For example test.map saved as test.map and test2 saved as test2.map.

General Instructions:
Installation:
git clone https://github.com/BimgBrein-42069/snek-game.git
Executing the game:
1) To run in windows cmd – window + R to open the run menu, then type cmd.
Then change working directory to where the game is located.
cd game_directory # game_directory is the directory where the game is stored
Then type
Python game.py
2) To run in linux – ctrl + alt + T to open terminal, then change working directory to where the game is located.
cd game_directory # game_directory is the directory where the game is stored

Then type

```
python3 game.py
```

Future improvements:

This project is a good one if done for the purpose of learning pygame or just for passion , but as a game, it is severely unfinished. Artwork in a game is highly important. If it comes together with a properly working game, people will want more games of similar or better quality from the developer. Of course the game should work properly or people will call it a cyberpunk 2077 ripoff.

Therefore, the development of this game will be suspended for some days and time for development will be replaced with the time for practicing art skills.

Once development resumes, work will first be done on the editor and its own UI, so that the player can use his custom levels.

Then further polishing will be done as long as the game is unfinished.