Product Requirements

Team Bob Ndertusat

Brief problem statement

The objective of the Snake Game project is to create a code for an arcade-style game that follows classic gameplay mechanics. Players will move the snake using keyboard input and guide it to consume food items, progressively increasing its length. However, they must also navigate the snake carefully to avoid collisions with borders or its body. The ultimate goal is to attain the highest score achievable within the game.

System requirements

To be able to play our snake game, it is necessary to have an IDE (ex. Visual Studio Code, Pycharm etc.) supporting Python 3 or a newer version of the programming language. It is also needed to have the modules turtle, time, random and tkinter as tk imported.

Users profile

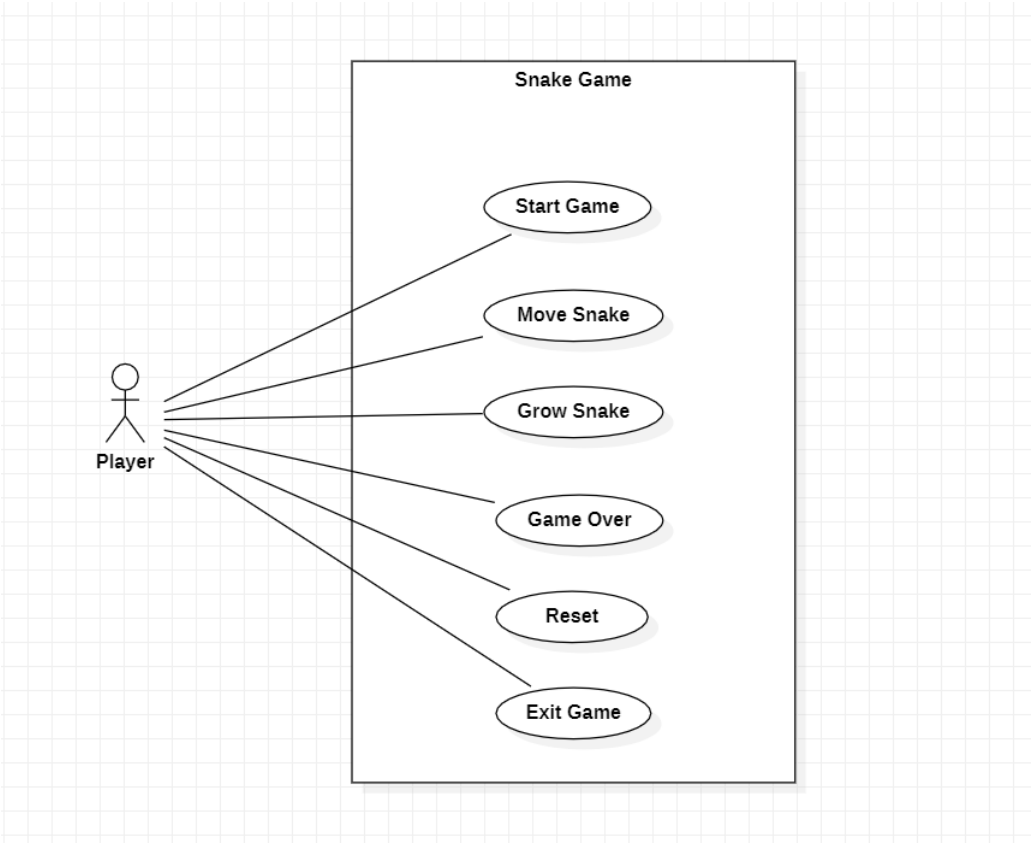
The initial release of the Snake Game is primarily aimed at young programmers who have a passion for gaming and seek an enjoyable activity during coding breaks. Our target audience comprises individuals with basic computer literacy and familiarity with keyboard controls, making it easy for them to engage with and have fun exploring our snake code.

Feature requirements (user stories)

No.	User Story Name	Description	Release
1.	Game board/Window	Creating the window where the game will be played. The moment that the code runs (starting the game) the window will open up to play the snake game. Users can stop playing by closing the window with the close button.	R1
2.	Snake	Creating the Snake with its initial body. Creating the function to move the snake's head and body throughout the open window in all directions (up, down, left, right).	R1
3.	Arrows	Create function to move the snake using the arrow keys	R1
4.	The keys "w", "d", "s", "a"	Create a function to move the snake using the keys "w", "d", "s", "a".	R1
5.	Food	Creating the food that will show up on the screen. Creating a function for the snake to eat food and creating a function to generate new food in random positions within the window boundaries each time.	R1
6.	Growing Snake	Create functions to grow the snake body each time it eats.	R1
7.	Scores	Create a function to keep track of current scores (no. of food eaten) and to print the update score live.	R1
8.	High Scores	Create a function to keep track of the Highest Score scored from the user on all the games played while running the program.	R1
9.	Colliding with borders	Creating a function for the snake to stop moving and the game to be over when the snake touches the window borders.	R1
10.	Snake eats itself	Creating a function for the snake to stop moving and for the game to be over when the snake 'eats itself' (the head crashes with the body).	R1

11.	Resetting/ Restarting	Creating a function to reset the game to its initial condition. The snake will be back to its original size, the current score will be back to 0 and the food will pop up to the first position. The user will be able to restart the game.	R1
12.	Sound effects	Create a function to add a sound effect when the snake moves, eats food, eats itself or crashes with the borders.	R3
13.	Pausing	Create function for user to be able to pause/freeze the game and then to continue again where he left	R2
14.	Levels	Creating a function to increase the difficulty of the game whenever the snake grows up by speeding the game.	R1
15.	Multiplayer	Creating a function to allow the player to change between one player and two players	R3

Use case diagram



Use case description

Use Case Number:	UC-01
Use Case Name:	Start Game
Overview:	Initialize the window where the game will be played. The snake, the food and the score will be displayed on the window
Actor(s):	Player
Pre condition(s):	The code should be opened in an IDE supporting Python 3 and the game should be running
Scenario Flow:	**Main (success) Flow:** 1- Setting up window (size, color, title) 2- Launching the window to start the game 3- Updating window content (snake, food, scores) 4- Start playing the game
	Alternate Flows: Exit Game (UC-06) 1- Run the code 2- Launch the game window 3- Play the game 4- Click on close button in the window to exit the game
Post Condition:	**Main Flow:** The game has started. The user can now play with the snake. The snake, the food and the scores are displayed on screen. **Alternate Flow:** The player is back at IDE looking at the main code after closing the window for the snake game.

Use Case Number:	UC-02
Use Case Name:	Move Snake
Overview:	Setting up the snake's initial head and body. Making a function for the movement of the snake by arrow keys or “w”, “s”, “d”, “a” keys to play the game and display everything on the window. Updating the screen each time the snake moves.
Actor(s):	Player
Pre condition(s):	The code should be running and the game must have started. The window is open and updated all the time.
Scenario Flow:	**Main (success) Flow:** 1- Start the game. 2- Window is opened and updated. 3- Display the snake on its initial condition and position. 4- Start moving the snake.
	Alternate Flows: Game Over (UC-04) 1- User is playing the game. 2- User is moving the snake. 3- The snake collides with an obstacle (snake body or borders)

	4- The snake stops moving. 5- Game is over.
Post Condition:	**Main Flow:** After the player moves the snake, the new position is updated on the window. The window will be updating the same state until the move is changed. **Alternate Flow:** After the player has crashed the snake with an obstacle the snake will stop moving. The player can either restart/reset or stop playing.

Use Case Number:	UC-03
Use Case Name:	Grow Snake
Overview:	The player moves the snake towards the randomly generated food's direction. The snake will eat the food and start growing each time. The score of the snake will also increase by representing the number of food eaten each time and the snake will start to speed up.
Actor(s):	Player
Pre condition(s):	The player should have the code running and using the keyboard input to move the snake towards the food.
Scenario Flow:	**Main (success) Flow:** 1- Move the snake Using keyboard keys. 2- Direct the snake towards the food. 3- The snake eats the food and increases the score. 4- The snake body grows.
	Alternate Flows: Reset (UC-05) 1- The snake is moving using keyboard input. 2- The snake collides with its body or the window borders. 3- The snake stops moving and growing. 4- Player restarts the game. 5- Snake body shrinks to initial state and the score resets to zero.
Post Condition:	**Main Flow:** The player is playing the game and increasing their score by growing the snake and making it eat the food. **Alternate Flow:** The player is back to having a small snake and no scores. The player needs to start growing the snake.