NAMAL INSTITUTE

Namal Institute Mianwali
Computer Science Department
Artificial Intelligence
Fall 21

Document:	**AI Project**	**Date:**	**06-12-2021**
Prepared by:	**Jawad Ahmed (BSCS-2019-59)** **Danish Khan (BSCS-2019-50)** **Muhammad Anas (BSCS-2019-49)** **Osama**	**Prepared for:**	**Mr.Faisal Zeeshan**
Document Details:	**This document contains the explanation of how Snails Game Works. How MiniMax and Heuristic Algorithms Work.**		

Tables Of Contents:

❖ Abstract:

- The significance of games is developing due to the reality that games are playing very interestingly worldwide especially in the young generation, and in this regard the companies prefer to develop more and more interesting games to engage the interesting one. Nowadays, the game industry produces maximum games based on Artificial Intelligence. Gaming on computers is an important part of the fast expanding game sector. While creating and constructing a game has traditionally been a commercial enterprise, we feel that it may also be a helpful project for teaching students about modelling and visual effects. Students are particularly exposed to the actual challenges involving animation, Geometric modelling, rendering, etc. AI based games are more interesting because both humans and bot(computer) have capability to think, move and decide who plays more efficiently.

- Keeping in mind the importance of AI based games in industry, we are designing a 2P grid based snail Game. The important approach is that we use AI in this 2P human and bot game to make bot artificially intelligent as it can beat humans.

1. Introduction:

- In Computer Science, game development is regarded as the major and hottest field.
- In this project, we develop 2D Snail computer games in which we are working on a grid. This game is about covering the grid space by two Snail players. This game can be playable by both human to human and human to bot snail. It is important to any player to cover maximum space in the grid to win this game. But the main purpose of the game is to make bot so intelligent that it can beat human snails. In order to make bot more intelligent we use different algorithms like minmax and heuristic search to find the optimal path.

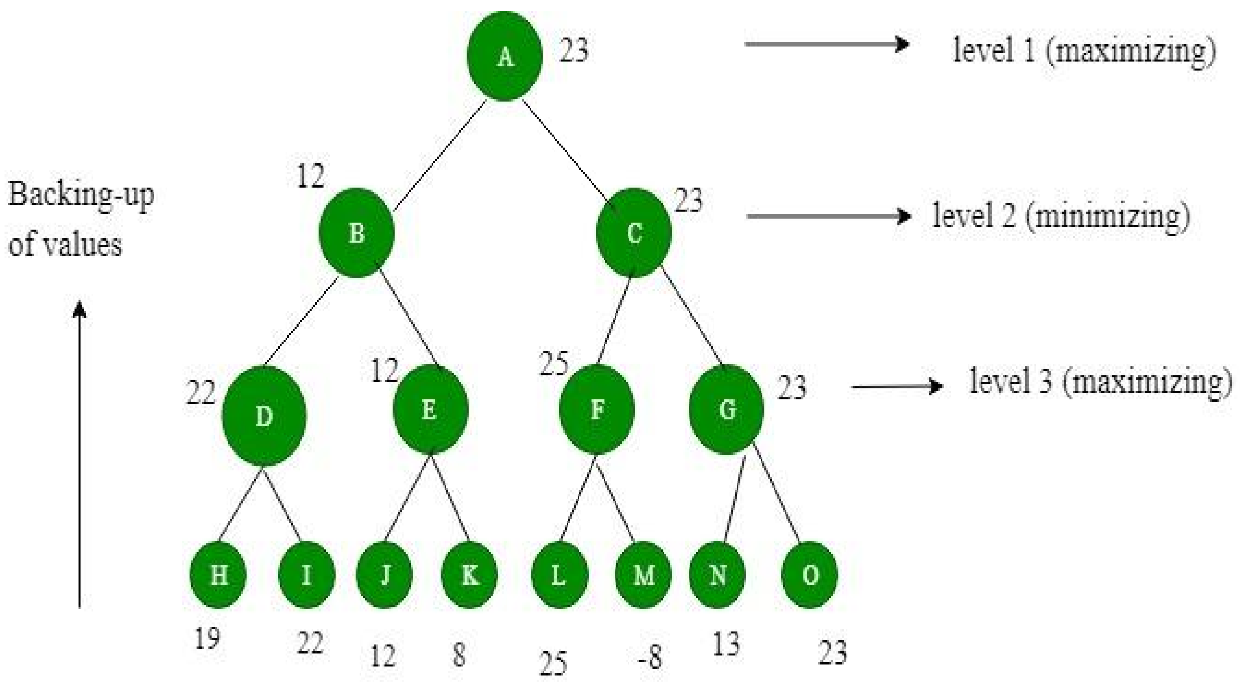
2. How to play game?

- It is important for any player to occupy the most spaces in the grid to win a game.
- In this game there is a fixed turn 1st move from human side and then bot and so one.
- The player can move his snail horizontally or vertically to an adjacent empty grid. And it will be a legal move and the player will be rewarded with a plus score.
- When any player moves his Snail to an empty grid, the snail leaves behind his splashes which will show it occupying the space.
- It's the player's choice whether he moves to an empty box or goes back from his occupied part.
- A player can’t move diagonally, it's an illegal move.
- When a player plays an illegal move, its turn will be lost and the opponent player's turn will do its turn.
- Turn will be lost if you click on opponents Snail or Splash.
- When almost all the grid space is occupied, at that time the game winner will be decided based on the more occupied space in the grid which means the highest score will be declared as winner.
- When the player moves on its slippery surface then its score will not increase. Only occupying empty spaces will increase its score.

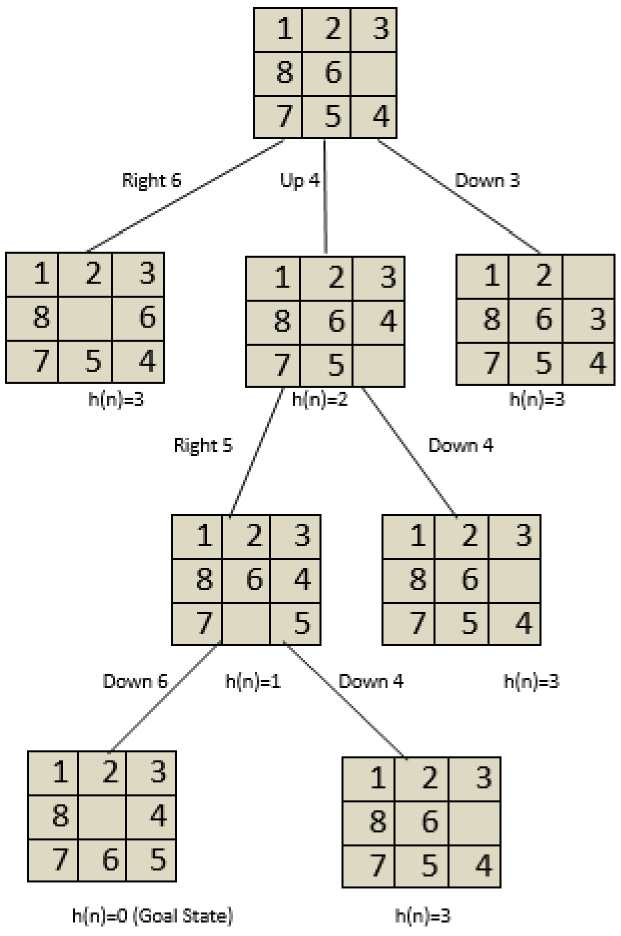
3. Literature Review:

- Literature review of this report tells us about the different functions used to solve the problem. In this report our problem is to make the bot intelligent so that it can beat the human player. To solve this problem we used different search techniques like the minimax algorithm. It will give us the optimal path but to make it more efficient and optimal we use the heuristic algorithm which helps the bot to think rationally as human mind.
- ○ The explanation of all algorithms used in this game are explained below:

- **MINI-MAX Algorithm:**
    - In Artificial intelligence the main component for problem solving is search algorithm
    - The Minimix algorithm helps the bot player to find all possible paths and gives the best optimal path to maximize the winning chances or minimize the losing chances. So in the minimax algorithm we define two players named min and max. So max will try to add score by +1 while on the other hand, min will try to add score by -1. It gives the best optimal path on the basis of values at terminal nodes meaning that its function is the same as DFS in which it will go upto leaf node and by the method of recursion it returns back, this behaviour of algorithm called as backtracking algorithm. With the help of information provided by the backtracking algorithm it gives the optimal path to the player.



- **Heuristic Algorithm:**
    - Heuristic algorithm is different from the minimax algorithm, as in the minimax algorithm all possible movements are exhausted and then it gives the best optimal path, while the heuristic algorithm limits the minimax algorithm and gives the best optimal path on the basis of predicted value. It does not search all the possible moves extensively but works more efficiently and fast because of providing the optimal path on the basis prediction instead of searching the all possible moves.



- **Alpha Beta Pruning:**
    - Another efficient search technique is Alpha–beta pruning, that aims to reduce the number of nodes in its search tree that are assessed using the minimax algorithm. This is an adversarial search method that's often used to play two-player games by machines (Tic-tac-toe, Chess etc.). It stops examining a move when it discovers at least one alternative that indicates it is worse than a previously considered move. Such actions do not need to be re-evaluated. When applied to a conventional minimax tree, it produces the same result as minimax, but prunes out branches that have no bearing on the ultimate decision.

4. Methodology/ Implementation:

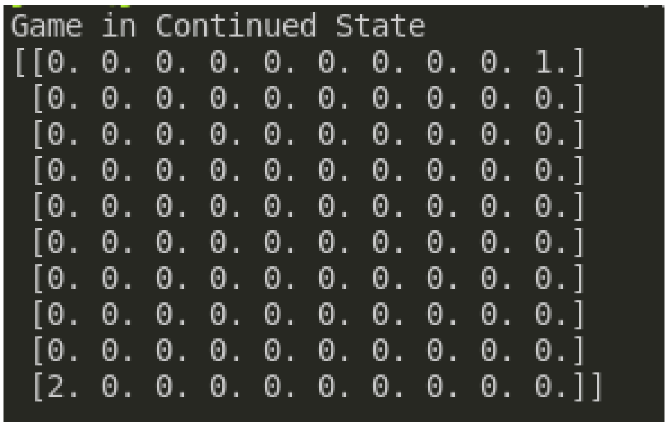
4.1 Language Used:

- The game is developed using python language which is more flexible and easy to use for this game.
- The IDE used for this game is VS code.

4.2 Game Development Library

- In python language, arcade is an important, easy and simple library for 2D game development. It is ideal for people learning to program, or developers that want to code a 2D game without learning a complex framework.So due to this flexibility we develop our game in Arcade python.

4.3 Functions Used In the Game:

- **Menu View Class:**
    - In this class, there is a welcome screen.
    - The draw function in this class will show, ‘Welcome to Snails Game’ and ‘Click Anywhere Will switch to rules page.
    - The mouse press function will shift the current window to the Game View Class.
- **Game View Class:**
    - Init function will initialize the variables that will be used later in the game.
    - **Grid Initialization:**
        - In this function the 10 by 10 grid will be initialized by the numpy library of python.
        - We also set the value of human (red-player) as 2 on the left bottom and the value of bot (silver-player) as 1 on the right top.
        - 

```
Game in Continued State
[[0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [2. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

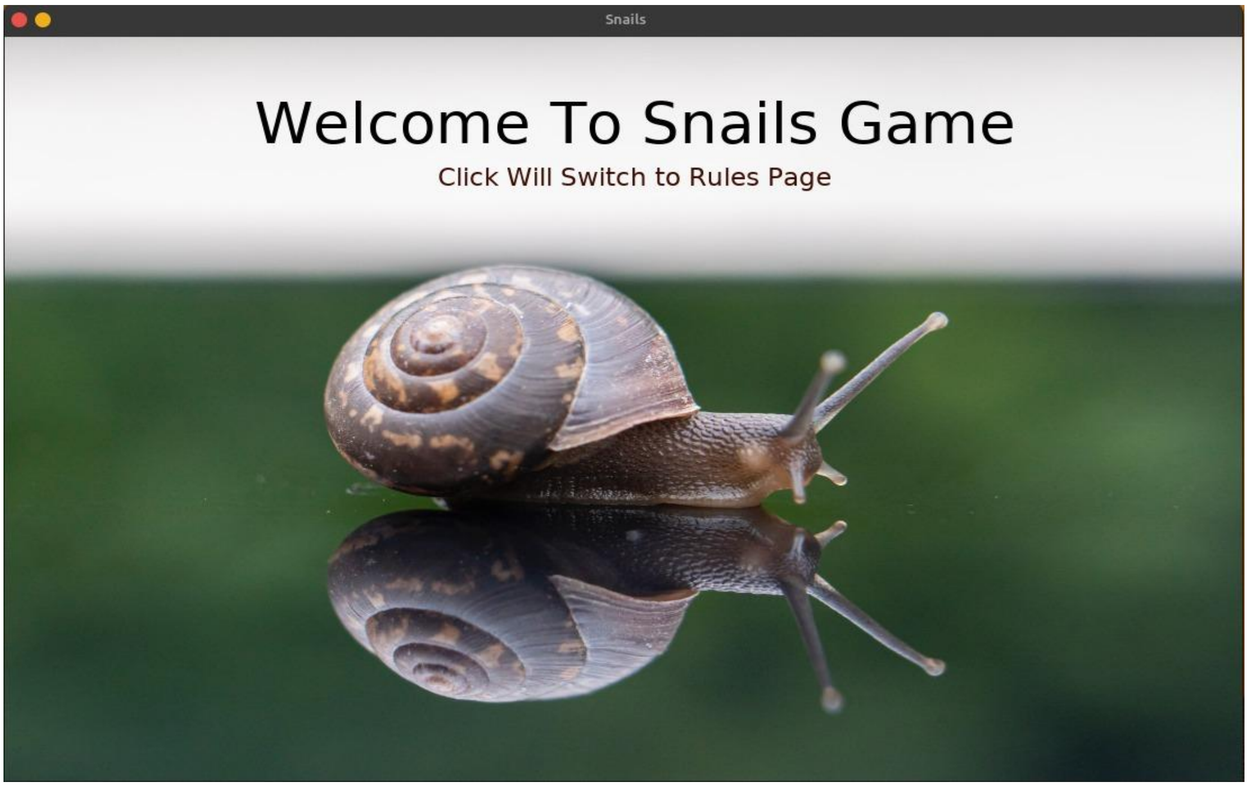
- **ON_Draw Function:**
    - There are three states in this function i.e Instruction Menu, Game On and Game End.
        - In the instruction menu there are rules defined and when we press space key it will shift to another state i.e GameOn.
        - In GameOn state background image is loaded and shown. Snails Logo along with title ‘Snails Game’ is also drawn on the right side of the Game. On the right side we also showed the status of which player turned and what is the score of that player.
        - We also draw the horizontal and vertical lines which are the frontend grid and shown as lines on the screen.
        - The most important work in the GameOn is the synchronization of the backend grid with the frontend grid and also placing the Red Player Sprite and its Splash and Silver Player Sprite and its Splash.
        - There is also a GameEnd state in which we draw the text ‘Draw’ , ‘Bot win’ and ‘Human win’. This state is called when the human or bot won the game or drew the game under certain conditions.
- **ON_Press Mouse Function:**
    - In this function, it will check the player turn and set its value in the grid.
    - It will also evaluate the board and will return the score.
    - In the function the isLegal and the slippery functions will be called in order to ensure the valid movement of sprites.
- **Is_Legal Move Function:**
    - This function will ensure the movement of sprites on its adjacent boxes in the grid.
    - It takes the current row, current column and the player turn and will return the status like 1 and left move etc. If the move is invalid then it will return False.
    - There are certain restrictions on the movement of snails, that is, snails can only move to their adjacent place and fill the previous covered place with a splash. So the snail has only the option to move right, left, down and up. Also no two snails can move to the same place or block at a time.
- **Slippery Functions:**
    - There are four slippery functions i.e left,right, up and down. As per the rules of the game the snails can only be on its back trail means the path is covered, so it can move back on that path. So these functions will do the part of back movement on the path followed.

- **MINIMAX Function:**
    - There are two players, mina and maxi, who play from their own perspectives. The mina try to minimize the opponent's winning chances by selecting the best move against the maxi and same with the maxi.
    - This function is called recursively and when the base condition is satisfied then it will return the best move against the player.
    - This minimax algorithm can be optimized by many other techniques like heuristic algorithms.
- **Heuristic Function:**
    - This function is used for making judgments quickly and efficiently.
    - In this function we pass the position and column and row and it will return the optimal position to move the bot.
    - This function is called by the minimax in order to optimize the minimax algorithm.
- **findBestmove Function:**
    - This will return the best possible move for the bot. In this function we call the minimax function.
    - It will traverse all cells, evaluate minimax function for all empty cells. And return the cell with optimal value and position.
- **botMovement Function:**
    - In this function, the findBestMove function will call from which the optimal value of column and row is returned and now it's the bot that will set according to the return values.
- **evaluateScore Function:**
    - In this function we check out the score by evaluating the board. It will return 0 if the state is continued, it will return 200 if the Bot scores more than the human player, similarly it will return 100 if the human scores more than the bot.

5. Game User Interface:

5.1 Welcome Screen:

- So, the first step when we run the code is the welcome screen as shown below and when you click on the welcome screen it goes to the instruction screen.

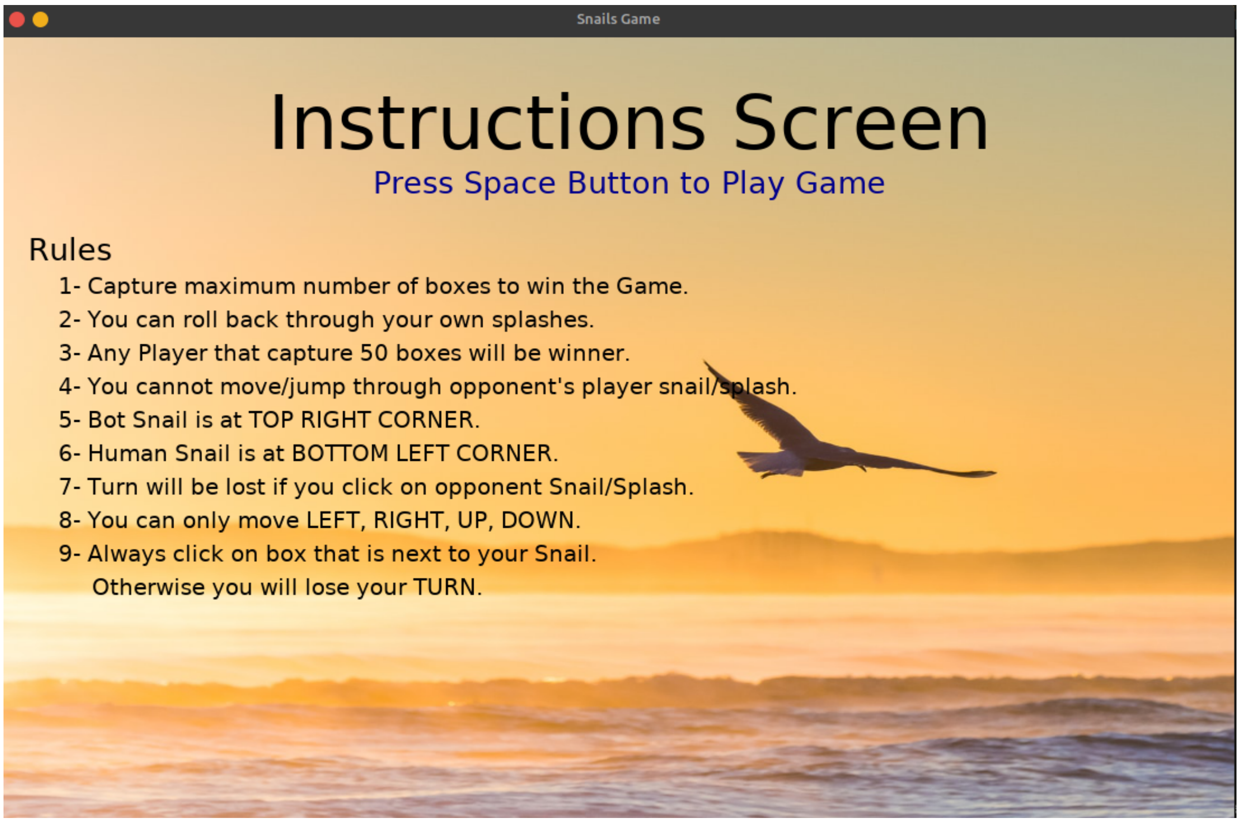


Welcome To Snails Game

Click Will Switch to Rules Page

5.2 Instruction Screen:

- When we click on the welcome screen, it will take you to the instruction screen. The Instruction window appears as seen in the picture below. Here when we press the Space key then it will take you to the game screen.



Instructions Screen

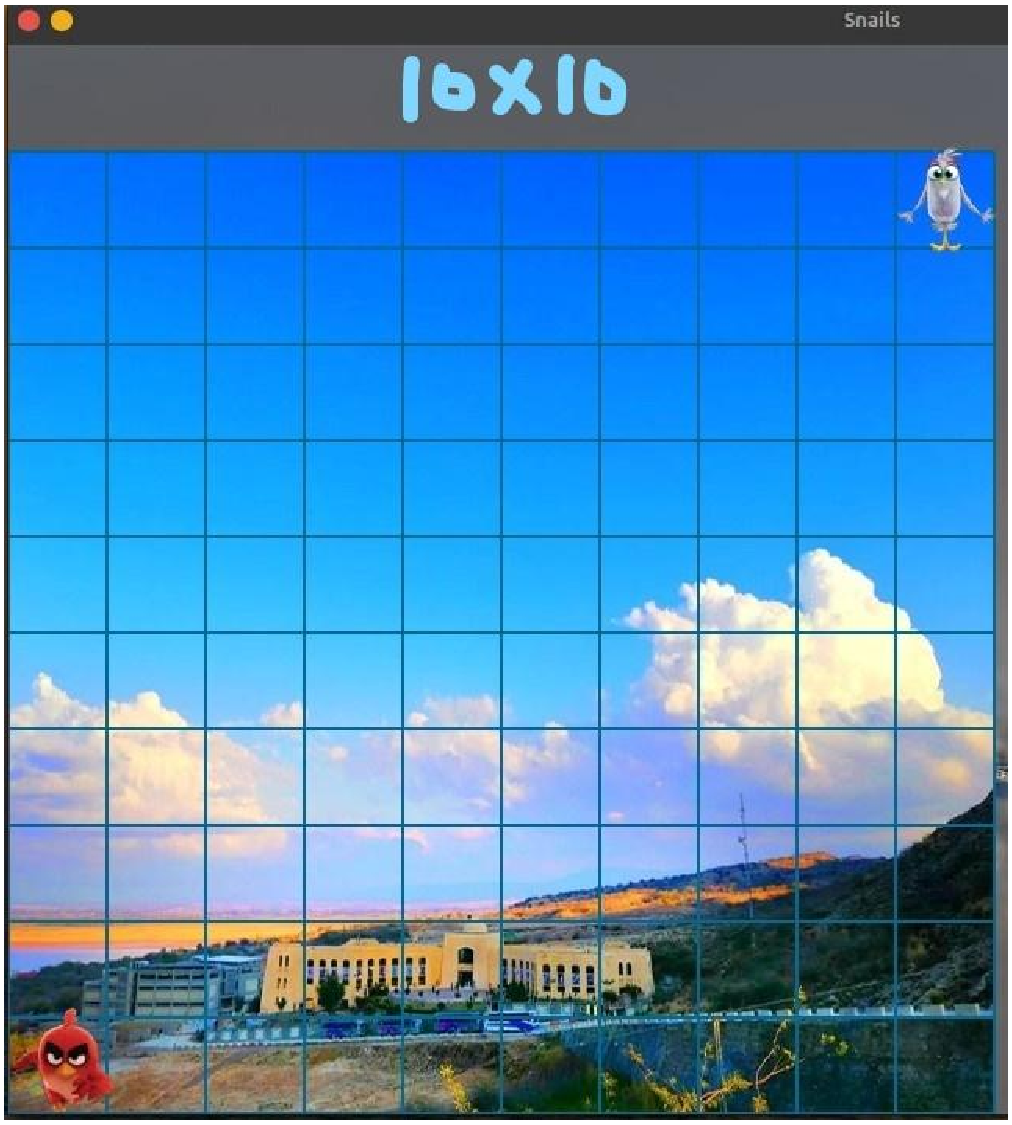
Press Space Button to Play Game

Rules

1- Capture maximum number of boxes to win the Game.
2- You can roll back through your own splashes.
3- Any Player that capture 50 boxes will be winner.
4- You cannot move/jump through opponent's player snail/splash.
5- Bot Snail is at TOP RIGHT CORNER.
6- Human Snail is at BOTTOM LEFT CORNER.
7- Turn will be lost if you click on opponent Snail/Splash.
8- You can only move LEFT, RIGHT, UP, DOWN.
9- Always click on box that is next to your Snail.
Otherwise you will lose your TURN.

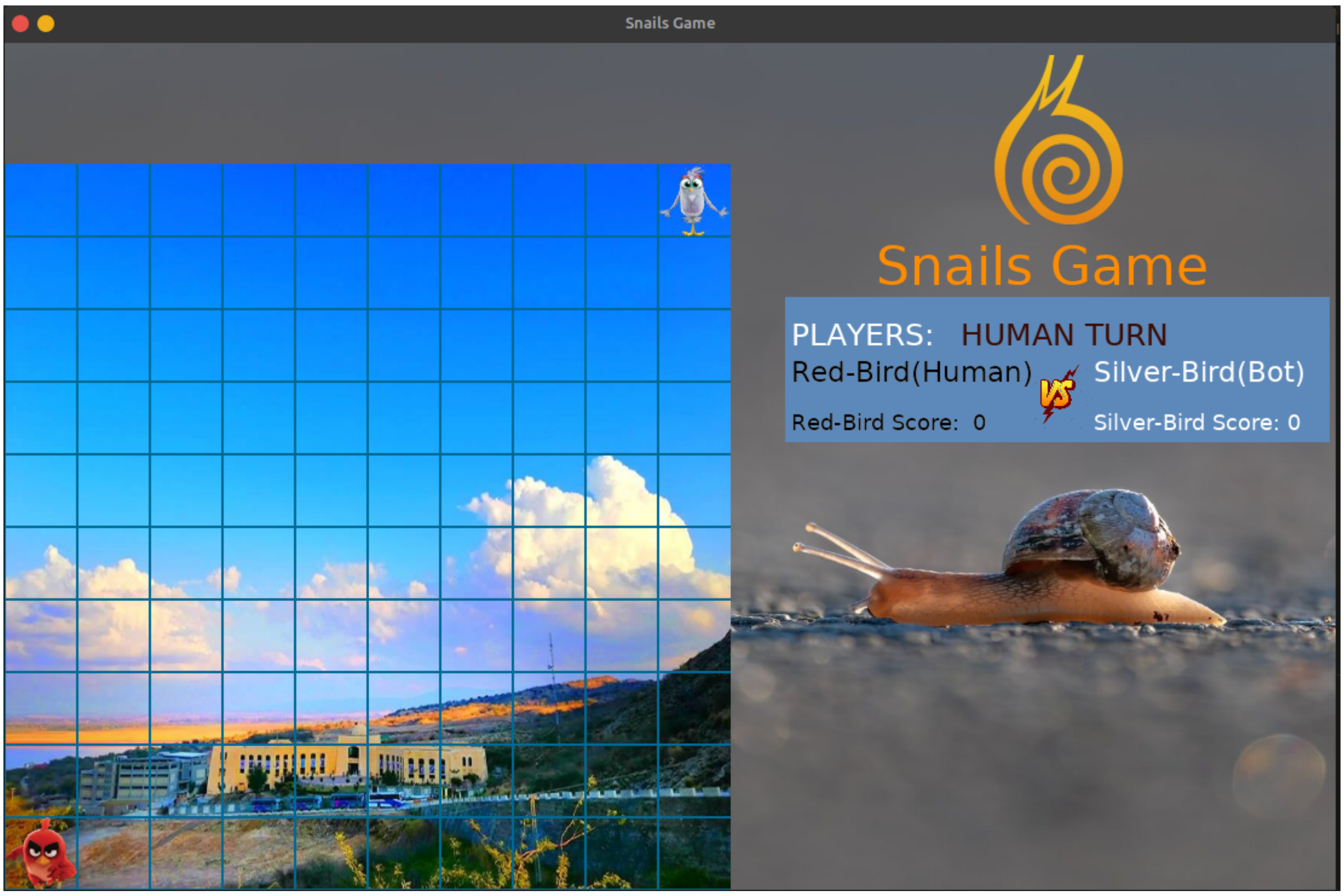
5.3 (10 by 10 Grid):

- Grid consists of 10 rows and columns which means the grid has 100 blocks. Initially we have to place both players at two different positions like one snail Bot placed at the top most right corner(0,9) block while the other at bottom left corner(9,0). As grid updates its values according to players movements so change in snail position leads to updation in grid values. As you can see in the picture below.



5.4 Game Screen:

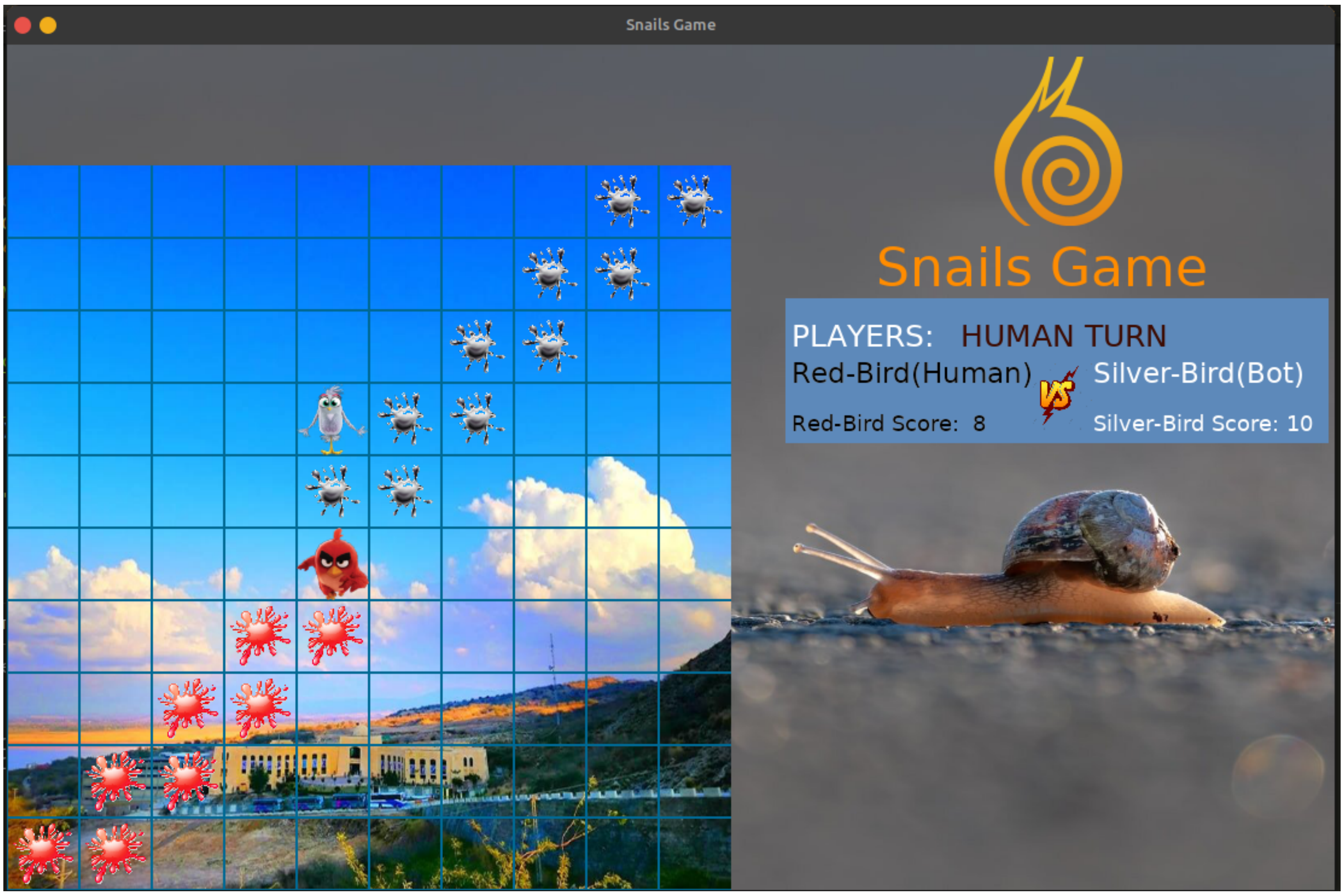
- The game screen is shown below:



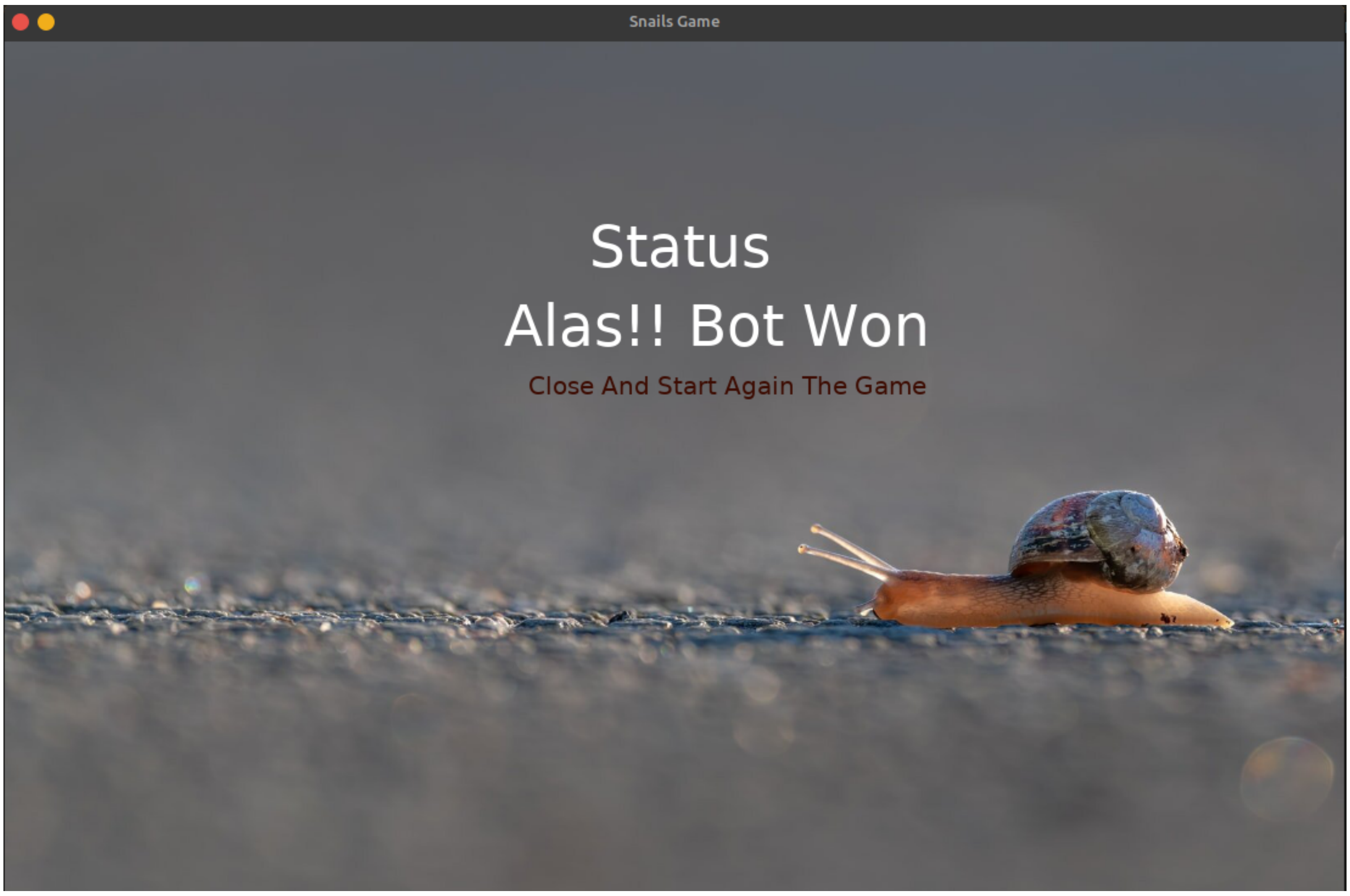
6. Testing and analysis:

- So, at the end of the game we have three situations that are win, lose or draw. So as discussed above that the player which cover the most boxes in grid will win the game, if both players covered equal number of boxes then game will draw or player with less number of covered boxes will lose.

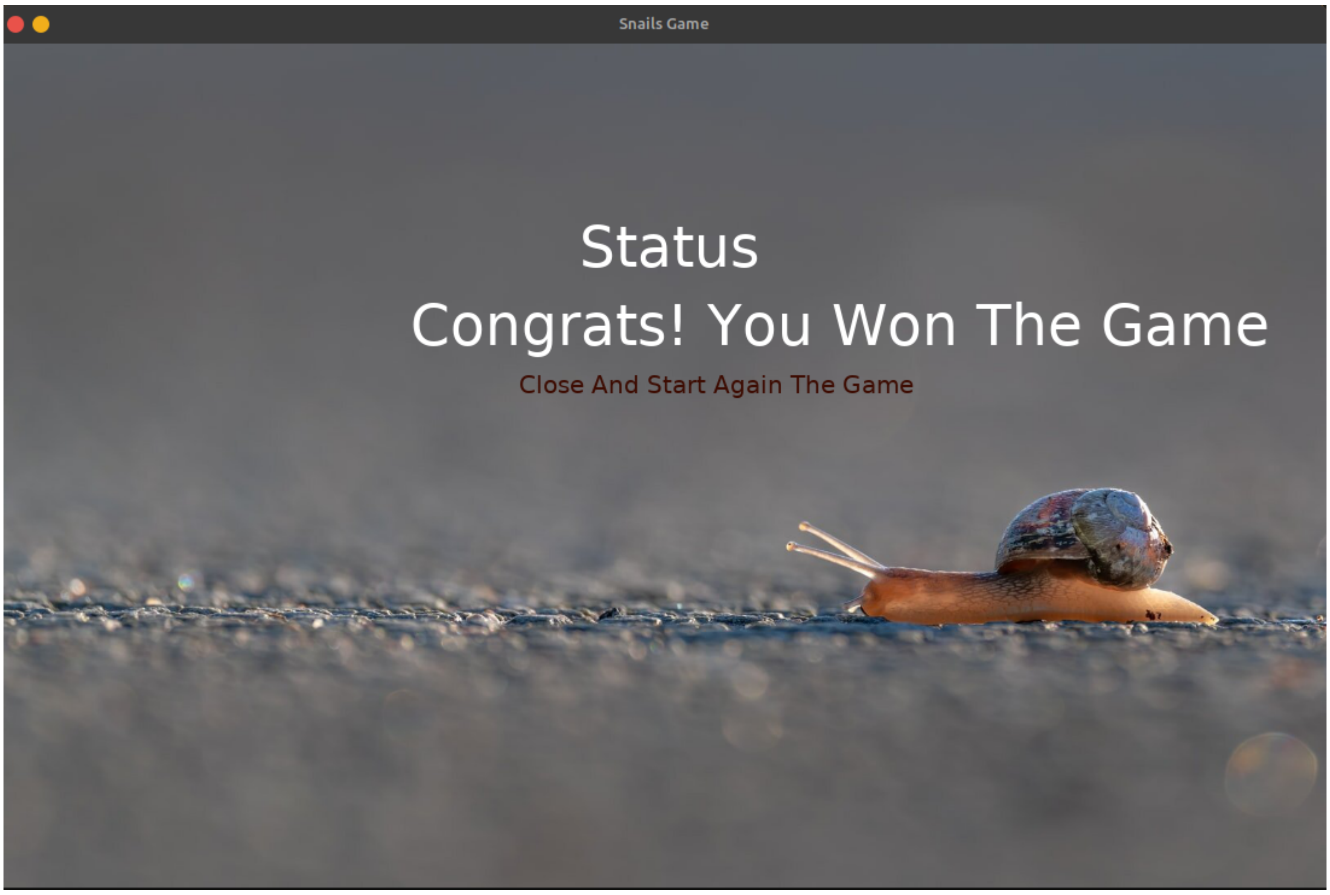
- After human bot will move as shown below:



- When Bot Won the game the following screen is shown:



- When human Won the Game the following windows is shown:



Snails Game

Status

Congrats! You Won The Game

Close And Start Again The Game

7. **Limitations of Game:**
    - This game can only be played with a mouse not keyboard.
    - This game is only playable on pc not on mobile.
    - Bot is not optimized that much so that can always win from the human
    - Game is played one time, for playing again you have to close it.
    - This game can be played on a grid having 10 rows and 10 columns.

8. **Conclusion:**

    - This is a simple game compared to chess and go games. minmax can't find optimal moves alone. For complex games like chess there are alpha-beta pruning, heuristic search and other algorithms along with minmax. When all these algorithms apply in complex games along with minmax in chess game then the moves strategy works efficiently.

9. References:

- https://en.wikipedia.org/wiki/Alpha%E2%80%93beta_pruning
- https://api.arcade.academy/en/latest/
- https://www.mygreatlearning.com/blog/alpha-beta-pruning-in-ai/