**PYTHON PROJECT**

***SNAKES N LADDERS***

(Group 5 Team 2)

**(a)Abstract:**

*>Objective*

1) Implement Snakes and ladders (with a few additional features) game using python. The additional feature is the inclusion of flags and points.

2) There are two types of flags viz. red and green. These flags are randomly generated on the board. The tile containing a red flag takes away 3 points from the player. Similarly a green flag awards the player 5 points.

3) Completing the game first i.e. reaching the 100th check first gifts 10 points.

4) The game ends as soon as one of the players reaches 100th check. Then their scores are compared and player with higher points is declared as the winner.

5)In case of tie in points, The player who reaches the 100th square first wins.

*>Method*

1) The project involves making a game using python. So, 'pygame' has been used.

2) The flag generation and dice roll are completely random. So, 'random' has been used.

3) To generate interest a music track is included. So, music tracks were downloaded

*>Result*

This allowed us to create a new version of snakes and ladders, which gives the player who hasn’t reached top check a fair chance to win.

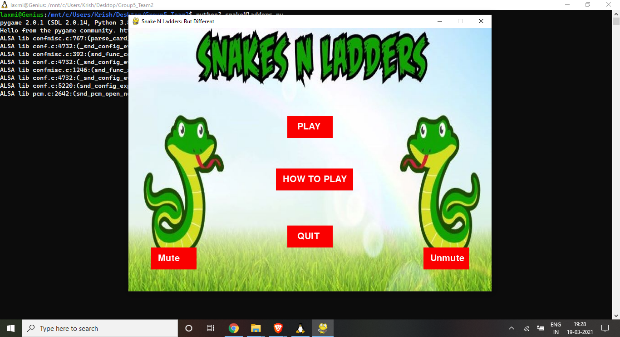
**(b)Introduction:**

>*Menu Screen*

1) The 'play' button: This button navigates the players to the board.

2) The 'How to play' button: Navigates user to instruction page.

3) The 'quit' button: This button exits the game.



>*Game Screen*

1) Has a background with white space in the middle which is surrounded by a multi-coloured diamond. The white space is mostly covered by the game board.

2) The top right corner of the game screen has a blue space where the dice roll is displayed. Below that there is a roll dice button, which when pushed, rolls the dice.

3) The game screen has a small area reserved to display the updates of the game. It informs the user when the dice is rolled, ladder is climbed or a snake eats them.

4) The bottom right corner has a space reserved to show the colour of the 'piece' which represents the player.

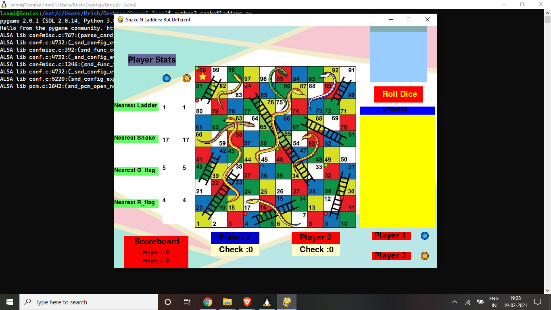
5) The bottom left has a score board which updates as the game progresses.

6) The left half is reserved to display player stats. Intimates the player about the nearest snake, nearest ladder, closest flags.

7) The game also displays the check numbers on which players are currently present.

8) The flags are represented by small circles which are at the top left corner of their respective checks.

9)The functionality of snakes and ladders is very similar to conventional snakes and ladders game.

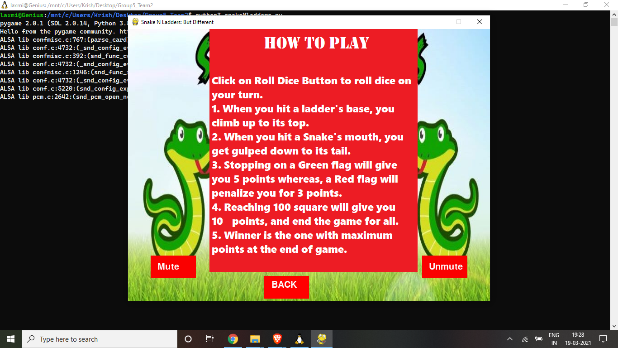


>*Instructions*

There is a designated Guide window which explains the game to a beginner.

>*Unmute & Mute*

They mute and unmute music depending upon the players’ choice.



**(c)System Requirements:**

1) The environment must be LINUX or LINUX- like.

2) 'Python3' has to be installed in the system.

3) 'Pygame' has to be installed in the system.

**(d)Work Division:**

*Kedar Deshpande (IMT2020523)-*

Implementing basic snakes and ladders game (rolling the dice, ladder, snake, etc), moving pieces accordingly (in a proper direction for any row), implementing the winner page, and buttons to mute and unmute background music.

*Asmita Zjigyasu (IMT2020507)-*

Implementing sounds which include background music, sounds that play if a player climbs a ladder, slides down a snake, or lands on red or green flags, sound track which plays when the winner is declared.

*Saket Gurjar(IMT2020520)-*

Implementing buttons such as PLAY, HOW TO PLAY, QUIT, and PLAY AGAIN, implementing full page of “How to play” with rules and guidelines and a BACK button.

*Laxmi Sreenivas (IMT2020510)-*

Implementing the display features such as player stats (nearest snake, ladder, flags), player positions as a number, text updates about the game, and indicating the player to roll the dice by changing colour of rectangle.

*Kaushik (IMT2020137)-*

Implementing the flags which include displaying them on the board (at random positions) and tracking the points of both the players accordingly and deciding a winner.

**(e)Flowchart:**

1) Player 1 starts the game by rolling the dice. The piece moves the according to the random number generated on the dice. If a 6 is generated, the player gets to roll the dice once again. If a 6 is generated again, the player moves 6 positions and again receives a chance. Once he/she gets a non-6 number, his/her token moves the place s as per the dice and the chance is transferred to player 2.

2) Player 2 rolls the dice and moves its position according to the number on the dice.

3) The chance keeps on getting transferred alternatively, keeping in mind the extra chance awarded by the number 6 on the dice.

4) Whenever a player encounters a snake, it slides down the board. Similarly, whenever a player encounters a ladder, it climbs up the board.

5) When a player encounters a red flag, 3 points are deducted and a green flag awards 5 points. These points are updated on the scoreboard.

6) The game ends as soon as a player reaches the 100th square. The player is awarded 10 points. The winner is decided based on the total accumulated points. The player with higher points is the winner. If both players have equal scores, the one which reaches 100 first, wins.

7) The game can be played again by clicking on the PLAY AGAIN button on the winner screen.





**(f)Self rendering board:**

**Stage 1: Creating a Board**

* There 600 x 600 space reserved for the board => The no. of checks rendered on the board is directly proportional to the width of each tile. So, by altering the width of tile in game we are indirectly altering the board size.
* To create the checks we need to iterate x with respect to a y from (0,600) with the step value equal to 'width'. Similarly need to iterate y from (0,600) with the step value equal to 'width'
* The colour of the tile can be altered based on the value of (x+y)%2.(mod operation with 2 only gives to possible values i.e. 0,1 so easy to handle)

**Stage 2: Generating Co-ordinates of the Tiles**

* Generating the coordinates of the tiles is important as we need them to construct the snakes and ladders.
* Generating them is a bit tricky as every alternate row the numbers on the tiles reverse their order. So while generating co-ordinates we need to them into make them into group them into members of ten. Members of every alternate group has to be reversed.

**Stage 3: Making the Ladder**

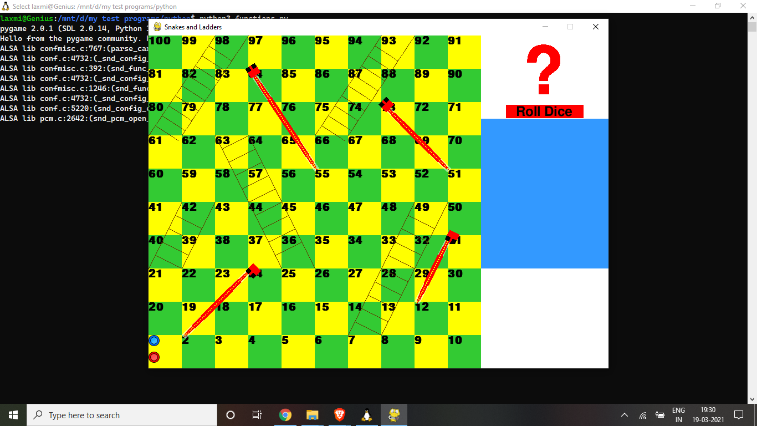
* To make the ladder form check A to check B.
* We need to join top left and top right corners of both the checks. So, now we have only 2 lines.
* The next step is to select one of the lines. Then get points on the line at certain intervals. After that drop the perpendiculars from these points onto the other line.
* Make sure the points you choose are sufficiently far enough.

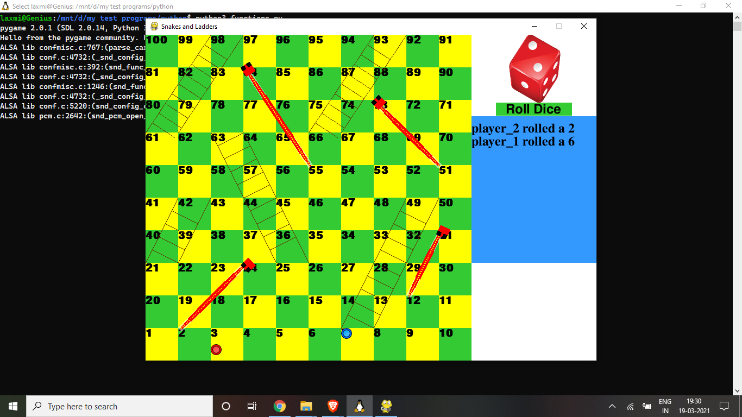
**Stage 4: Making the Snake**

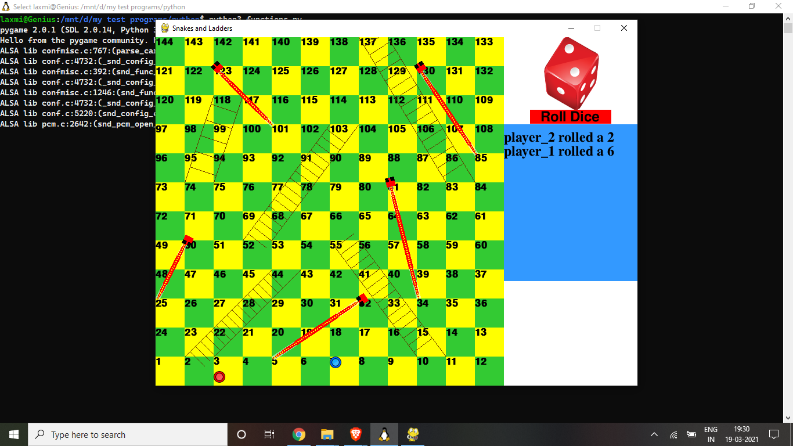
* Let the join check A and check B. The length of the surface must be distance between the tiles.For the breadth any small number would do.
* Then make a surface at (min(Ax,Bx), min(Ay,By)).
* Drawn an ellipse at (0,0) of the surface. This becomes our snakes tail.
* For the snakes head we make a rectangle at right most end of the ellipse. You can make two more smaller rectangles in the snakes head to represent its eyes.
* Finally, find theta which is tan-1(y2-y1/x2-x1) where (x1,y1) ; (x2,y2) are co-ordinates of check A and check B
* Rotate the surface at angle of -theta.

The features of the board are very similar to that of normal snakes and ladders but the board size increases as the time progresses altering the pattern of snakes and ladders present.









**(g)Future Aspect:**

1) By increasing board size, this game can be converted form 2 player to multi-player

2) A quick quiz can be made as a tie breaker. Improving players knowledge

3) A game against AI can be implemented.

**(h)Reference:**

1)https://www.pygame.org/docs/

2)<https://stackoverflow.com/>