APPENDIX 1

SNAKES AND LADDERS

PROJECT REPORT

by

BHANUPRATAP , HIMANSHU GANDHI

(Section:K19QW)

(Roll Number(s): B60 , B63)

Department of Intelligent Systems

School of Computer Science Engineering

Lovely Professional University, Jalandhar

OCT-2020

APPENDIX 2

Student Declaration

This is to declare that this report has been written by us. No part of the report is copied from other sources. All information included from other sources have been duly acknowledged. We aver that if any part of the report is found to be copied, we are shall take full responsibility for it.

<<HIMANSHU GANDHI>>

<<Roll number: B63 >>

<<BHANUPRATAP>>

<<Roll number: B60 >>

Place : Jalandhar

Date :30/10/2020

APPENDIX 3

TABLE OF CONTENTS

TITLE

APPENDIX 4

BONAFIDE CERTIFICATE

Certified that this project report “ SNAKES AND LADDERS ” is the bonafide work of “ HIMANSHU GANDHI and BHANUPRATAP ” who carried out the project work under my supervision.

<<Name of supervisor>>

<<Academic Designation>>

<<ID of Supervisor>>

<<Department of Supervisor>>

AIM: To make a snakes and ladders using dice rolling simulator of python

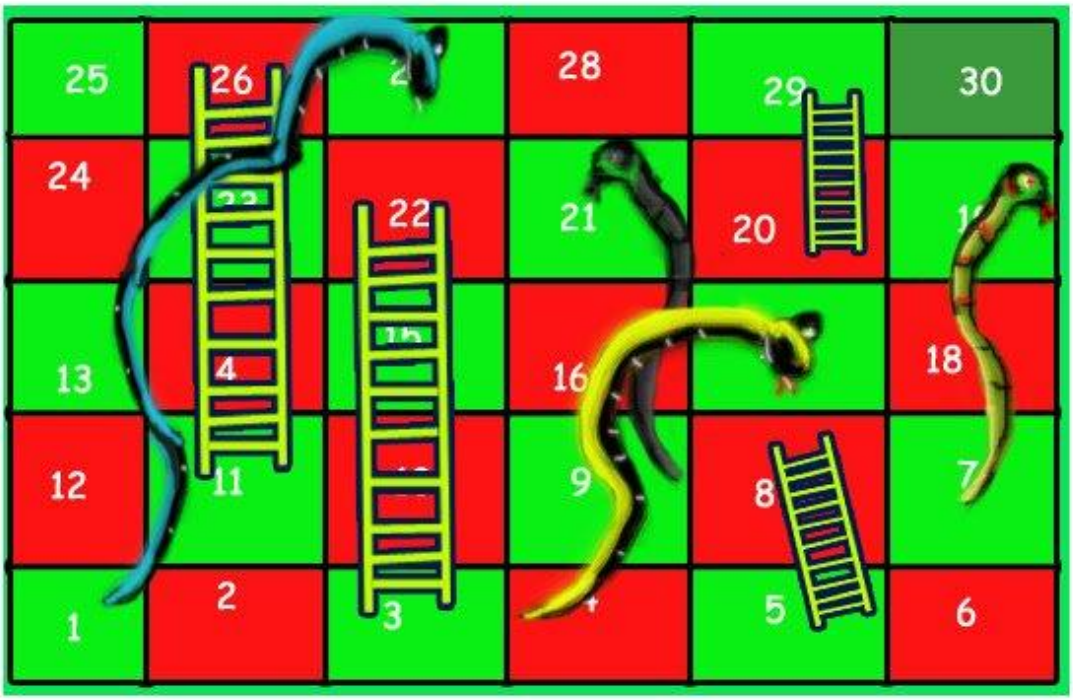
BRIEF DESCRIPTION :

Python has the ability to create snakes and ladders by using the dice rolling simulator of python . It is also very simple to use. It along with numpy and other python built-in functions achieves the goal. In this article we will see some of the different kinds of numbers dice can generate.

- ⇨ **ABOUT :**

If the player reaches a cell which is base of a ladder, the player has to climb up that ladder and if reaches a cell is mouth of the snake, has to go down to the tail of snake without a dice throw.

For example, consider the board shown, the minimum number of dice throws required to reach cell 30 from cell 1 is 3.
Following are the steps:
a) First throw two on dice to reach cell number 3 and then ladder to reach 22
b) Then throw 6 to reach 28.



c) Finally through 2 to reach 30.

The idea is to consider the given snake and ladder board as a directed graph with number of vertices equal to the number of cells in the board. The problem reduces to finding the shortest path in a graph. Every vertex of the graph has an edge to next six vertices if next 6 vertices do not have a snake or ladder. If any of the next six vertices has a snake or ladder, then the edge from current vertex goes to the top of the ladder or tail of the snake. Since all edges are of equal weight, we can efficiently find shortest path using Breadth First Search of the graph.
Following is the implementation of the above idea. The input is represented by two things, first is ‘N’ which is number of cells in

the given board, second is an array ‘move[0…N-1]’ of size N. An entry move[i] is -1 if there is no snake and no ladder from i, otherwise move[i] contains index of destination cell for the snake or the ladder at i.

⇨ TIME COMPLEXITY :

Time complexity of the above solution is O(N) as every cell is added and removed only once from queue. And a typical enqueue or dequeue operation takes O(1) time.

```
#Snakes and Ladders simulation

import random #importing the random function to be able to use it later on

counterposition = 0 #setting counterposition and diceroll to 0
currentDiceroll = 0

def diceroll (): #when user rolls the 1-6 dice this does it randomly
   return random.randint (1,6)
userInput = int(input("How many games would you like to play snakes and
ladders?"))

for i in range (userInput):
   currentDiceroll = diceroll


 currentDiceroll = diceroll()
   print("The currentDiceroll is", currentDiceroll)

   if counterposition == 1:                #all the if statements show what happens if
the one player lands on a snake or a ladder
      counterposition = counterposition + 37
   if counterposition == 4:
      counterposition = counterposition + 10
```

```
  if counterposition == 9:
     counterposition = counterposition + 22
  if counterposition == 21:
     counterposition = counterposition + 21
  if counterposition == 28:
     counterposition = counterposition + 56
  if counterposition == 51:
     counterposition = counterposition + 16
  if counterposition == 72:
     counterposition = counterposition + 19
  if counterposition == 80:
     counterposition = counterposition + 19
  if counterposition == 17:
     counterposition = counterposition - 10
  if counterposition == 54:
     counterposition = counterposition - 20
  if counterposition == 63:
     counterposition = counterposition - 4
  if counterposition == 64:
     counterposition = counterposition - 4
  if counterposition == 87:
     counterposition = counterposition - 51
  if counterposition == 92:
     counterposition = counterposition - 19
  if counterposition == 95:
     counterposition = counterposition - 20
  if counterposition == 98:
     counterposition = counterposition - 19
  if counterposition >= 100:
     print ("Congratulations end of game")

  counterposition = counterposition + currentDiceroll
  print("the counter position is", counterposition)
```

Name is the player's name
Current is the player's current position (an integer)
New is the player's new position after they moved. This is an integer if the player has not completed the game or a * if they have.

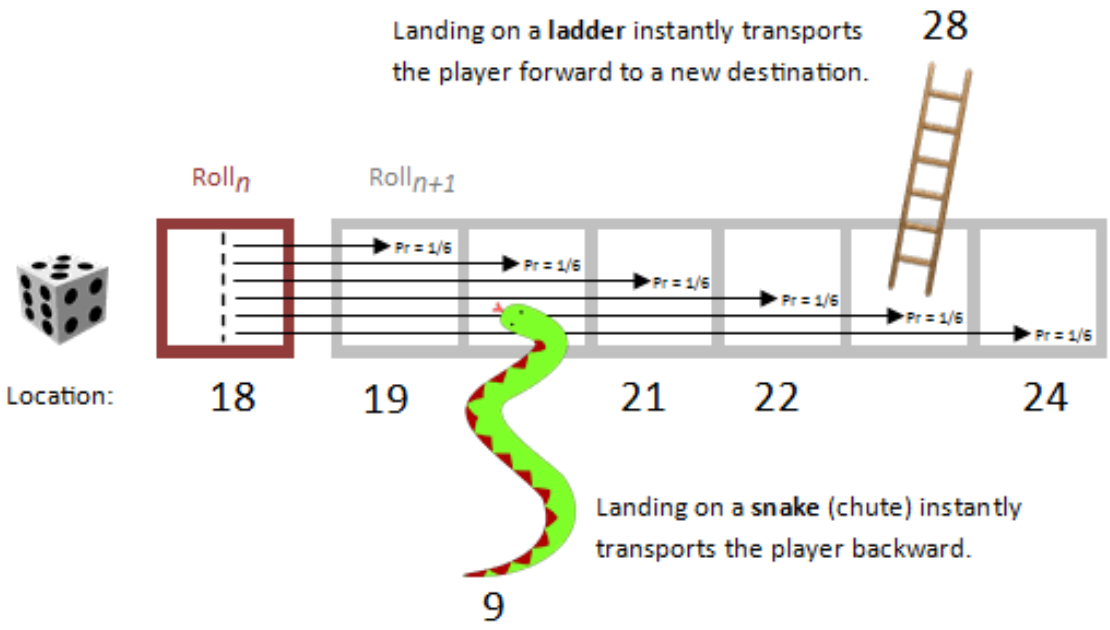
For example, if the game is being played on the board in the above figure, it is Alice's turn, she is on square 25, and she rolls a three, then the output for her turn would be:

```
Alice 25 84
```

更多

Example

Sample Input	Sample Output
10 4 SNAKE 9 1 LADDER 4 8 LADDER 2 5 SNAKE 7 3 3 Alice Bob Carol 9 3 5	Alice 1 8 Bob 1 6 Carol 1 3 Alice 8 * Bob 6 1 Carol 3 3 Bob 1 6 Carol 3 8 Bob 6 *



WORK DIVISION:

All the work was done equally by all the team members. Each task was divided in such a way that equal participation of all the team member is there.

1) Collection of resources and content material
2) Coding and testing
3) Feedback and implementation
4) Preparation of project report

PLATFORM USED :

ANACONDA PYTHON

SWOT ANALYSIS:

The primary aim of developing this snakes and ladders is to provide a source of entertainment to the one who plays it😊.

GitHub Link: https://github.com/HimanshuGandhi/SNAKES-AND-LADDERS