**Snake Game**

****

***Dissertation Submitted in Partial fulfillment of the***

***Requirement for the Award of the Degree of***

***Master of Computer Application (6 Yrs.) IV Semester***

## Session - Jan-May, 2019

**Under the guidance of Submitted By**

Kirti Mathur Name-Rohan Telang

Roll No.-IC-2K17-82

**International Institute of Professional Studies**

**Devi Ahilya Vishwavidyalaya, Indore, M.P.**

**2019**

**DECLARATION**

I hereby declare that the project entitled ”Classic Snake Game“ submitted by Rohan Telang (IC-2K17-82)for the partial fulfillment of the requirement for the award of Master of Computer Application (6 Years) IV Semester to International Institute of Professional Studies, Devi Ahilya Vishwavidyalaya, Indore, comprises my own work and due acknowledgement has been made in text to all other material

used.

Signature of Student:



Date: 07/05/2019

Place: Indore

**CERTIFICATE**

It is to certify that we have examined the project on “Classis Snake Game”, submitted by Rohan Telang to the International Institute of Professional Studies, DAVV, Indore and hereby accord our approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfillment for the award of the degree of “Master of Computer Application (6 Years) IV Semester”.

**Internal Examiner External Examiner**

Signature**:** Signature**:**

Name  **:** Name **:**

Date **:**………………Date **:**……………..

## ACKNOWLEDGEMENT

I acknowledge my sincere thanks to those who have contributed significantly to this project. It is a pleasure to extend deep gratitude to our internal guide Kirti Mathur, Director of IIPS for their valuable guidance and support and to continuously prompt us for the progress of the project. I thank them for their valuable suggestions towards our project, which helped us in making this project more efficient and user friendly.

I thank and acknowledge each and every ones efforts that helped me in some or the other way for small and significant things.

Student Signature –



Student Name – Rohan Telang

Roll No – IC-2K17-82

**ABSTRACT**

This project aims to bring the fun and simplicity of snake game with some basic features. It will include computer controlled intelligent opponents whose aim will be to challenge the human players. It will also store high score of the player and challenge the other players.

This project explores a new dimension in the traditional snake game to make it more interesting and challenging. The simplicity of this game make it an ideal candidate for a minor project as we can focus on advanced topics like multiplayer functionality and implementation of computer controlled intelligent opponents.

**Table of Contents**

* **Declaration**
* **Certificate**
* **Acknowledgement**
* **Abstract**

1. **Introduction** 
   1. The Client Organization
   2. Problem Definition
   3. Aim
   4. Objectives
   5. Projects Goals
   6. Benefits
2. **Analysis** 
   1. Questionnaire
   2. Statistical Analysis
3. **Project Planning**
   1. Project Scope
   2. Document Plan
   3. Team Structure
   4. Gantt Chart
4. **System Design**
   1. Logical Design
      1. Entity Definition
      2. Attribute Definition
      3. Relationships
      4. E-R Diagram
      5. Data Flow Diagram
      6. Flow Chart
      7. Use Case Diagram
   2. Physical Design
      1. User Interfaces
5. **System Development Methodology**
6. **System Implementation**
7. **System Testing**
   1. Black Box Testing
   2. White Box Testing
   3. Unit Testing
   4. Integration Testing
   5. System Testing
8. **Output Forms & Reports**
9. **Limitation**
10. **Conclusion**

Introduction

# The client organization -

Clients of this game are gamer.

# Problem definition -

It will help people to increase their IQ level. It will also improve your sensitivity.

# Aim –

This project aims to bring the fun and simplicity of snake game with some basic features. It will include computer controlled intelligent opponents whose aim will be to challenge the human players.

**Objectives -** This game aims to change the way people think of traditional snake game. It will offer the  experience of commercial multilayer games to the player retaining the simplicity of traditional  snake game. The major objectives of this project are:

● Create a snake game that will have all the functionality of traditional snake games.

● Introduce multilayer functionality in the game that will allow several players to play a game simultaneously. It should be able to give the experience of a real time multiplayer game to the players.

● Introduce computer controlled intelligent opponent (unique feature of this game) to make the game more challenging and interesting. The movement and action of these intelligent opponents will be controlled by computer whose aim will be to eat the food before human players capture it.

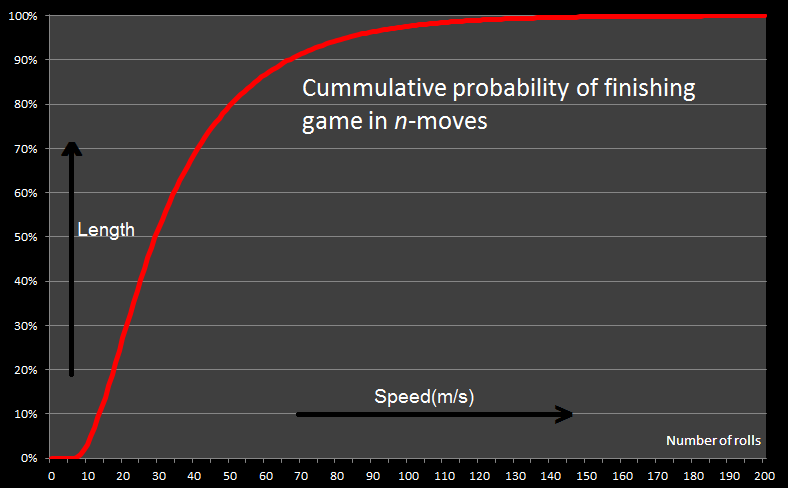
# Projects Goals –

The main **goal** of this **game** is to feed an increasing length of a **snake** with food particles which are found at random positions, picking up bonus mongooses that occur at regular intervals.

# Benefits –

It will improve player’s IQ level and sensitivity

Analysis



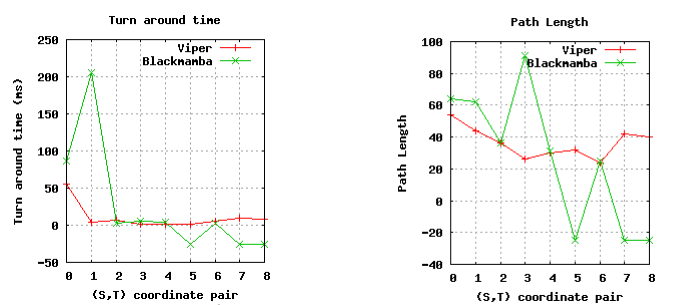
As the length of the snake increases the speed of the snake increases i.e. Length is directly proportional to speed. The food will be placed randomly on the plane. Snake will with ASWD keys. When snake grabs the food, length will increase. It will show the high score. As it grab the food, the score will increase by 10 points. When snake will bits its own body or it collide the wall, the game will be over.

These are computer controlled snakes, in the game, whose aim is to challenge the human players. We have two implementations of path finding algorithms to create intelligent autonomous opponent snakes. These algorithms return the shortest possible path from given source (S) and target (T) coordinate pair considering the obstacles (if any) present in the game field. The code name3 for these two implementations are:

● Blackmamba (refer to ANNEX  A)

● Viper (refer to ANNEX  B)

A detailed paper describing the algorithm used by these two implementation is present in ANNEX A and ANNEX B. To know which of the two implementations perform better, we profiled them using a simple JUnit test and the results were plotted using gnuplot:



Turnaround time plot The time elapsed between the instant of supplying the (source, target) coordinate pair (S,T) to the algorithm and the instant when it returns a path for supplied (S,T) pair is called turnaround time. From the plot of illustration 3, it is clear that the turnaround time for Viper is always smaller as compared to Black mamba. Path Length plot the length of path (computed by counting the number of game field coordinates in the path) returned by the two path finding algorithms is depicted by the plot of illustration 4. It is clear from the plot that Viper implementation results in smaller paths (and hence efficient) as compared to Black mamba implementation. NOTE: The value of turnaround time and path length for the coordinate pairs 5,7,8 have negative values for Black mamba. This suggests that the algorithm was not able to compute a path for given (S,T) pair in given timeout period (250 ms for this test).

Project Planning

# Project Scope –

The main work is to write a snake game using python programming language. This game will generate various problems related to mathematics, player has to answer the questions by controlling a snake with keyboard keys [A, W, D, S] to eat the answer ball. Each correct answer ball eaten prolongs the snake’s life.

# Source Code –

import turtle

import time

import random

delay = 0.2

speed\_increaser = 10

#speed= 0.1

score = 0

high\_score = 0

file= open("high\_score.txt",'r')

high\_score=int(file.readline())

file.close()

wn = turtle.Screen()

wn.title("Snake Game")

wn.bgcolor("green")

wn.setup(width=600, height=600)

wn.tracer(0)

head = turtle.Turtle()

head.speed(0)

head.shape("circle")

head.color("black")

head.penup()

head.goto(0, 0)

head.direction = "stop"

food = turtle.Turtle()

food.speed(0)

food.shape("circle")

food.color("red")

food.penup()

food.goto(0, 100)

segments = []

pen = turtle.Turtle()

pen.speed(0)

pen.shape("square")

pen.color("white")

pen.penup()

pen.hideturtle()

pen.goto(0, 260)

pen.write("Score: 0 High Score: "+ str(high\_score), align="center", font=("Courier", 24, "normal"))

def go\_up():

if head.direction != "down":

head.direction = "up"

def go\_down():

if head.direction != "up":

head.direction = "down"

def go\_left():

if head.direction != "right":

head.direction = "left"

def go\_right():

if head.direction != "left":

head.direction = "right"

def move():

if head.direction == "up":

y = head.ycor()

head.sety(y + 20)

if head.direction == "down":

y = head.ycor()

head.sety(y - 20)

if head.direction == "left":

x = head.xcor()

head.setx(x - 20)

if head.direction == "right":

x = head.xcor()

head.setx(x + 20)

wn.listen()

wn.onkeypress(go\_up, "w")

wn.onkeypress(go\_down, "s")

wn.onkeypress(go\_left, "a")

wn.onkeypress(go\_right, "d")

while True:

wn.update()

if head.xcor() > 290 or head.xcor() < -290 or head.ycor() > 290 or head.ycor() < -290:

time.sleep(1)

head.goto(0, 0)

head.direction = "stop"

for segment in segments:

segment.goto(1000, 1000)

segments.clear()

delay = 0.2

speed\_increaser = 10

score = 0

delay = 0.2

pen.clear()

pen.write("Score: {} High Score: {}".format(score, high\_score), align="center", font=("Courier", 24, "normal"))

if head.distance(food) < 20:

x = random.randint(-290, 290)

y = random.randint(-290, 290)

food.goto(x, y)

new\_segment = turtle.Turtle()

new\_segment.speed(0)

new\_segment.shape("square")

new\_segment.color("grey")

new\_segment.penup()

segments.append(new\_segment)

delay -= 0.001

score += 10

if score > high\_score:

high\_score = score

pen.clear()

pen.write("Score: {} High Score: {}".format(score, high\_score), align="center", font=("Courier", 24, "normal"))

for index in range(len(segments) - 1, 0, -1):

x = segments[index - 1].xcor()

y = segments[index - 1].ycor()

segments[index].goto(x, y)

if len(segments) > 0:

x = head.xcor()

y = head.ycor()

segments[0].goto(x, y)

move()

for segment in segments:

if segment.distance(head) < 20:

time.sleep(1)

head.goto(0, 0)

head.direction = "stop"

for segment in segments:

segment.goto(1000, 1000)

segments.clear()

score = 0

if score == speed\_increaser and delay >= 0.010 :

delay = delay - 0.010

speed\_increaser=speed\_increaser+30

#print(speed\_increaser, delay)

pen.clear()

pen.write("Score: {} High Score: {}".format(score, high\_score), align="center",

font=("Courier", 24, "normal"))

file2= open("high\_score.txt",'w')

file2.write(str(high\_score))

file2.close()

time.sleep(delay)

print("end")

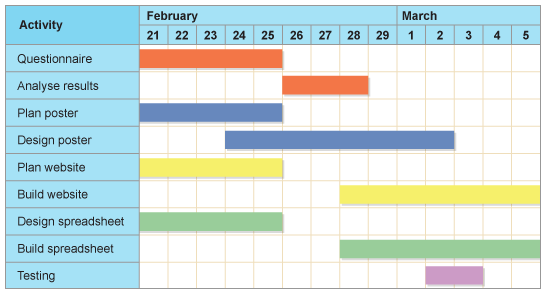
wn.mainloop()

Team Structure –

Rohan Telang – Development and Back End

Rahela Naaz – Designing and Front End

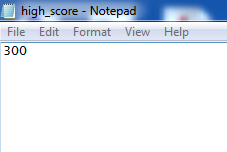
Gantt Chart –



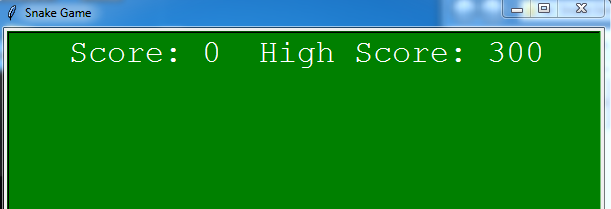
System Design

File system-

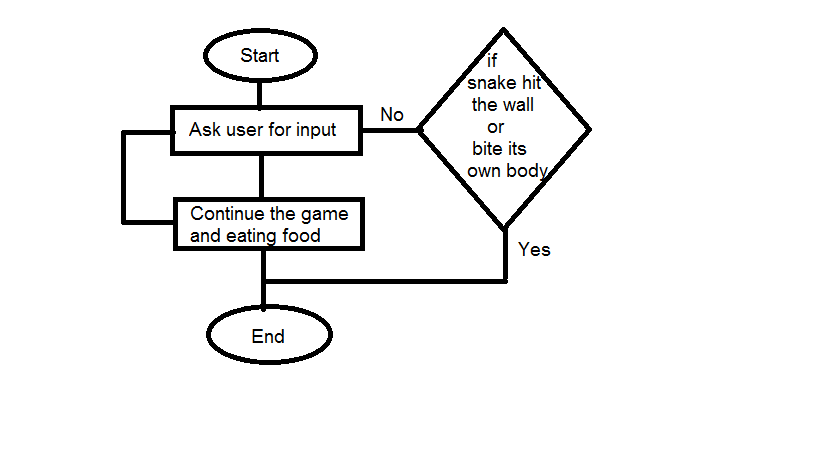
To store the score I use file handling. When user makes a new high score that will write in the text document:



When a new user will plays the game then it will be a challenge for him to beat that high score which will show on the game screen:

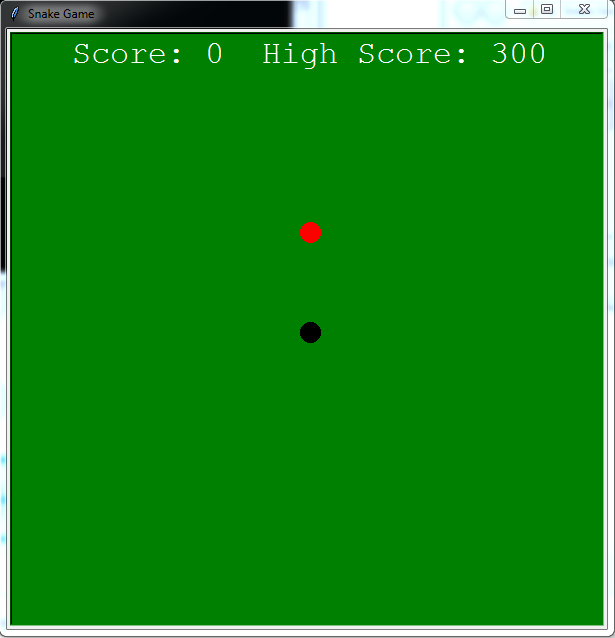


Flow Chart

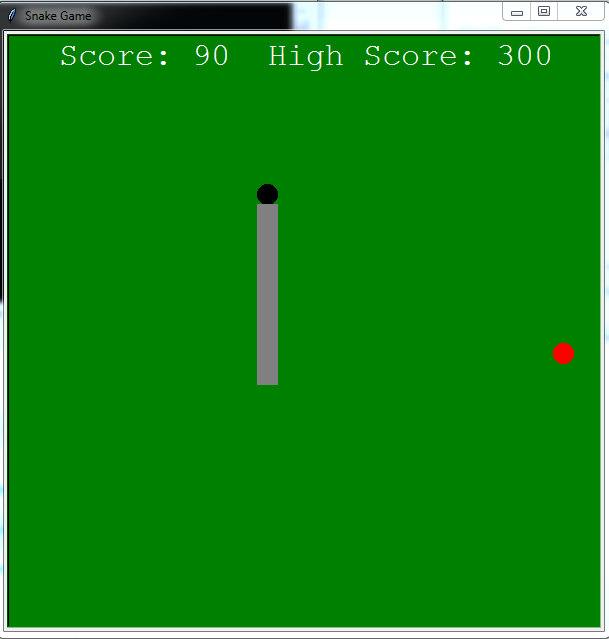


User Interface

Initial Screen –



During play –



System Development Methodology

* I made this game using Python with IDLE Python 3.
* I include Turtle, Time and Random Modules.
* Turtle is use for Front End and UI of the game.
* Time is used for time delay.
* Random is used to put food randomly.
* I use file handling to store the scores in the text document.

System Implementation

Firstly, I initialized the score=0 and high\_score=0. I made a variable for time delay i.e. delay=0.2.

Now using file handling implement the read and write of the high\_score. Then made a variable wn.turtle.Screen() for title of the game i.e. snake game, background colour and screen size.

Now made an another variable i.e. head=turtle.Turtle() for shape, colour and location of snake. Made an another variable i.e. food=turtle.Turtle() for shape, colour and location of food. I made an another variable pen=turtle.Turtle() for shape, color, initial location of body. As the length of the snake will increases then 10 points will add in scores. Then, define some functions def go\_up(), def go\_down(), def go\_left(), def go\_right() for initialization of keys. Then for movement of snake I define a function def move(). All this functions are in main loop i.e. wn.mainloop().

System Testing

Black-box testing

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance.

White-box testing

White-box testing is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases.

Unit testing

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use.

Integration testing

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements. It occurs after unit testing and before validation testing.

System testing

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements. System testing takes, as its input, all of the integrated components that have passed integration testing.

Limitation

If the number of FOR increases to too high, then it will obviously be tough and slow to determine the shortest path.

Conclusion

We were successful in creating a traditional version of snake game. The computer controlled intelligent opponents have been successfully tested in the game is a unique feature of Snake Game. We learned several project management techniques used by professionals to develop large scale project. The experience of working in team and integration of modules developed independently, with just requirement specifications, is a very important achievement for the Snake Game team.