SECOND YEAR B. TECH THIRD SEMESTER

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

MINI PROJECT REPORT(ITW-1)

ON

SNAKE GAME

By

ARNAV PATIL (BT21CSE078)

MANGALAM RAJ (BT21CSE091)

KALPESH SALAVE (BT21CSE127)

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY,

NAGPUR

(An Institution of National Importance by Act of Parliament)

BACKGROUND OF PROJECT: USE OF OOP , TURTLE MODULE AND RANDOM MODULE TO CREATE AN AMAZING SNAKE GAME !

METHODOLOGY:

CREATING DIFFERENT CLASSES FOR DIFFERENT FUNCTIONALITY AND COMPONENTS OF THE GAME

THE CLASSES ARE SEGREGATED IN DIFFERENT FILES FOR AUNTHENTIC ORGANIZATION

ALL THE CLASSES ARE IMPORTED IN THE MAIN CLASSES , WHERE EACH OF THEM IS CALLED ACCORDING TO IT'S USAGE

RESULTS:

THE SNAKE GAME IS SUCCESFULLY CREATED WITH LIVE SCORE AVAILABLE!

CODE :

Code for Snake file

```
from turtle import Turtle

STARTING_POSITION = [(0, 0), (-20, 0), (-40, 0)]


class Snake:
    def __init__(self):
        self.segments = []
        self.create_snake()

    def create_snake(self):
        for new_turtle in STARTING_POSITION:
            self.add_segment(new_turtle)

    def add_segment(self, new_turtle):
        turs = Turtle()
        turs.penup()
        turs.shape("square")
        turs.goto(new_turtle)
        turs.color("white")
        self.segments.append(turs)

    def extend(self):
        self.add_segment(self.segments[-1].position())

    def move(self):

        for positions in range(len(self.segments) - 1, 0, -1):
            new_xcor = self.segments[positions - 1].xcor()
            new_ycor = self.segments[positions - 1].ycor()
            self.segments[positions].goto(new_xcor, new_ycor)
        self.segments[0].forward(20)

    def turn_right(self):
        self.segments[0].right(90)

    def turn_left(self):
        self.segments[0].left(90)
```

Code for Pen file

```
from turtle import Turtle


class Pen(Turtle):
    def __init__(self):
        super().__init__()
        self.hideturtle()
        self.score = 0
        self.pencolor("white")
        self.penup()
        self.setpos(x=-20, y=260)

    def increased_score(self):
        self.score += 1
        self.clear()
        self.pendown()
        self.write(f"Score: {self.score} ", align="center", font=("areal",
24, "normal"))
```

Code for Food File

```
from turtle import Turtle
import random


class Food(Turtle):
    def __init__(self):
        super().__init__()
        self.shape("circle")
        self.penup()
        self.shapesize(stretch_wid=0.5, stretch_len=0.5)
        self.color("red")
        self.speed("fastest")

    def refresh(self):
        random_occur_x = random.randint(-260, 260)
        random_occur_y = random.randint(-260, 260)
        self.goto(random_occur_x, random_occur_y)
```

Code for main file

```
from turtle import Screen
import time
from snake import Snake
from food import Food
from pen import Pen

screen = Screen()
screen.listen()
screen.tracer(0)
screen.setup(width=600, height=600)
screen.bgcolor("black")
screen.title("Snake Game")

snake = Snake()
food = Food()
pen1 = Pen()
game_is_on = True

point = 0

while game_is_on:
    screen.update()
    time.sleep(0.1)
    snake.move()

    if screen.onkey(key="d", fun=snake.turn_right):
        snake.turn_right()
    elif screen.onkey(key="a", fun=snake.turn_left):
        snake.turn_left()

    if snake.segments[0].xcor() > 280 or snake.segments[0].ycor() > 280 or
snake.segments[0].xcor() < -280 or snake.segments[0].ycor() < -280:
        game_is_on = False
        print("GAME OVER")

    for segment in snake.segments:
        if segment == snake.segments[0]:
            pass
        elif snake.segments[0].distance(segment) < 10:
            game_is_on = False
            print("Game over")

    if snake.segments[0].distance(food) < 15:
        pen1.increased_score()
        food.refresh()
        snake.extend()
```

```
screen.exitonclick()
```