**SNAKE GAME USING TKINTER**

A PROJECT REPORT

*Submitted by,*

Soumya Prasad

ABSTRACT

This project is a classic Snake Game implementation using python and the Tkinter library. In this game, the player controls a snake that moves around the board to eat food and grow while avoiding obstacles and collision with the walls and its own body. If the snake touches itself or the border of the game then the game will over. In this project we define classes and function to implementing the game.

SOURCE CODE

from tkinter import\*

import random

window=Tk()

window.title("Snake Game")

score=0

direction="down"

game\_width=700

game\_height=700

speed=200

space\_size=30

Body\_parts=2

snake\_color="#8E7618"

food\_color="#DC143C"

background\_color="#FFFFC2"

l=Label(window,text="Score:{}".format(score),font="Arial 20")

l.pack()

c=Canvas(window,bg=background\_color,height=game\_height,width=game\_width)

c.pack()

class Snake:

def \_\_init\_\_(self):

self.body\_size=Body\_parts

self.squares=[]

self.coordinates=[]

for i in range(0,Body\_parts):

self.coordinates.append([0,0])

for x,y in self.coordinates:

square=c.create\_rectangle(x,y,x+space\_size,y+space\_size,fill=snake\_color,tag="snake")

self.squares.append(square)

class Food:

def \_\_init\_\_(self):

x=random.randint(0,(game\_width//space\_size)-1)\*space\_size

y=random.randint(0,(game\_height//space\_size)-1)\*space\_size

self.coordinates=[x,y]

c.create\_oval(x,y,x+space\_size,y+space\_size,fill=food\_color,tag="food")

def next\_turn(snake,food):

x,y=snake.coordinates[0]

if direction=="up":

y-=space\_size

elif direction=="down":

y+=space\_size

elif direction=="left":

x-=space\_size

elif direction=="right":

x+=space\_size

snake.coordinates.insert(0,(x,y))

square=c.create\_rectangle(x,y,x+space\_size,y+space\_size,fill=snake\_color)

snake.squares.insert(0,square)

if x==food.coordinates[0] and y==food.coordinates[1]:

global score

score+=1

l.config(text="Score:{}".format(score))

c.delete("food")

food=Food()

else:

del snake.coordinates[-1]

c.delete(snake.squares[-1])

del snake.squares[-1]

if check\_collisions(snake):

game\_over()

else:

window.after(speed,next\_turn,snake,food)

def change\_direction(new\_direction):

global direction

if new\_direction=="left":

if direction!="right":

direction=new\_direction

elif new\_direction=="right":

if direction!="left":

direction=new\_direction

elif new\_direction=="up":

if direction!="down":

direction=new\_direction

elif new\_direction=="down":

if direction!="up":

direction=new\_direction

def check\_collisions(snake):

x,y=snake.coordinates[0]

if x<0 or x>=game\_width:

return True

elif y<0 or y>=game\_height:

return True

for body\_parts in snake.coordinates[1:]:

if x==body\_parts[0] and y==body\_parts[1]:

return True

return False

def game\_over():

c.delete(ALL)

c.create\_text(c.winfo\_width()/2,c.winfo\_height()/2,fill="red",font=("consolas", 70),text='GAME OVER',tag="gameover")

window.update()

window\_width=window.winfo\_width()

window\_height=window.winfo\_height()

screen\_width=window.winfo\_screenwidth()

screen\_height=window.winfo\_screenheight()

x=int((screen\_width/2)-(window\_width/2))

y=int((screen\_height/2)-(window\_height/2))

window.geometry(f"{window\_height}x{window\_width}+{x}+{y}")

window.bind("<Left>",lambda event:change\_direction("left"))

window.bind("<Right>",lambda event:change\_direction("right"))

window.bind("<Up>",lambda event:change\_direction("up"))

window.bind("<Down>",lambda event:change\_direction("down"))

snake=Snake()

food=Food()

next\_turn(snake,food)

window.mainloop()

SCREENSHOTS



