```
#Snake Tutorial Python

import math
import random
import pygame
import tkinter as tk
from tkinter import messagebox

class cube(object):
    rows = 20
    w = 500
    def __init__(self,start,dirnx=1,dirny=0,color=(0,255,0)):
        self.pos = start
        self.dirnx = 1
        self.dirny = 0
        self.color = color



    def move(self, dirnx, dirny):
        self.dirnx = dirnx
        self.dirny = dirny
        self.pos = (self.pos[0] + self.dirnx, self.pos[1] + self.dirny)

    def draw(self, surface, eyes=False):
        dis = self.w // self.rows
        i = self.pos[0]
        j = self.pos[1]

        pygame.draw.rect(surface, self.color, (i*dis+1,j*dis+1, dis-2, dis-2))
        if eyes:
            centre = dis//2
            radius = 3
            circleMiddle = (i*dis+centre-radius,j*dis+8)
            circleMiddle2 = (i*dis + dis -radius*2, j*dis+8)
            pygame.draw.circle(surface, (0,0,255), circleMiddle, radius)
            pygame.draw.circle(surface, (0,0,255), circleMiddle2, radius)




class snake(object):
    body = []
    turns = {}
    def __init__(self, color, pos):
        self.color = color
        self.head = cube(pos)
        self.body.append(self.head)
        self.dirnx = 0
        self.dirny = 1

    def move(self):
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()

            keys = pygame.key.get_pressed()

            for key in keys:
                if keys[pygame.K_LEFT]:
                    self.dirnx = -1
                    self.dirny = 0
                    self.turns[self.head.pos[:]] = [self.dirnx, self.dirny]
                elif keys[pygame.K_RIGHT]:
                    self.dirnx = 1
                    self.dirny = 0
                    self.turns[self.head.pos[:]] = [self.dirnx, self.dirny]

                elif keys[pygame.K_UP]:
                    self.dirnx = 0
                    self.dirny = -1
                    self.turns[self.head.pos[:]] = [self.dirnx, self.dirny]

                elif keys[pygame.K_DOWN]:
                    self.dirnx = 0
                    self.dirny = 1
                    self.turns[self.head.pos[:]] = [self.dirnx, self.dirny]

        for i, c in enumerate(self.body):
            p = c.pos[:]
            if p in self.turns:
                turn = self.turns[p]
                c.move(turn[0],turn[1])
                if i == len(self.body)-1:
                    self.turns.pop(p)
            else:
                if c.dirnx == -1 and c.pos[0] <= 0: c.pos = (c.rows-1, c.pos[1])
                elif c.dirnx == 1 and c.pos[0] >= c.rows-1: c.pos = (0,c.pos[1])
                elif c.dirny == 1 and c.pos[1] >= c.rows-1: c.pos = (c.pos[0], 0)
                elif c.dirny == -1 and c.pos[1] <= 0: c.pos = (c.pos[0],c.rows-1)
                else: c.move(c.dirnx,c.dirny)
```

```
def reset(self, pos):
    self.head = cube(pos)
    self.body = []
    self.body.append(self.head)
    self.turns = {}
    self.dirnx = 0
    self.dirny = 1



def addCube(self):
    tail = self.body[-1]
    dx, dy = tail.dirnx, tail.dirny

    if dx == 1 and dy == 0:
        self.body.append(cube((tail.pos[0]-1,tail.pos[1])))
    elif dx == -1 and dy == 0:
        self.body.append(cube((tail.pos[0]+1,tail.pos[1])))
    elif dx == 0 and dy == 1:
        self.body.append(cube((tail.pos[0],tail.pos[1]-1)))
    elif dx == 0 and dy == -1:
        self.body.append(cube((tail.pos[0],tail.pos[1]+1)))

    self.body[-1].dirnx = dx
    self.body[-1].dirny = dy



def draw(self, surface):
    for i, c in enumerate(self.body):
        if i ==0:
            c.draw(surface, True)
        else:
                c.draw(surface)



def drawGrid(w, rows, surface):
    sizeBtwn = w // rows

    x = 0
    y = 0
    for l in range(rows):
        x = x + sizeBtwn
        y = y + sizeBtwn

        pygame.draw.line(surface, (255,255,255), (x,0),(x,w))
        pygame.draw.line(surface, (255,255,255), (0,y),(w,y))



def redrawWindow(surface):
    global rows, width, s, snack
    surface.fill((0,0,0))
    s.draw(surface)
    snack.draw(surface)
    drawGrid(width,rows, surface)
    pygame.display.update()



def randomSnack(rows, item):

    positions = item.body

    while True:
        x = random.randrange(rows)
        y = random.randrange(rows)
        if len(list(filter(lambda z:z.pos == (x,y), positions))) > 0:
            continue
        else:
            break

    return (x,y)



def message_box(subject, content):
    root = tk.Tk()
    root.attributes("-topmost", True)
    root.withdraw()
    messagebox.showinfo(subject, content)
    try:
        root.destroy()
    except:
        pass



def main():
    global width, rows, s, snack
    width = 500
    rows = 20
    win = pygame.display.set_mode((width, width))
    s = snake((0,0,255), (10,10))
    snack = cube(randomSnack(rows, s), color=(255,0,0))
    flag = True

    clock = pygame.time.Clock()
    while flag:
        pygame.time.delay(50)
        clock.tick(8)
        s.move()
        if s.body[0].pos == snack.pos:
            s.addCube()
            snack = cube(randomSnack(rows, s), color=(255,0,0))

        for x in range(len(s.body)):
            if s.body[x].pos in list(map(lambda z:z.pos,s.body[x+1:])):
                print('Your Score is : ', len(s.body))
                message_box('You Lost!', 'Play again.....')
                s.reset((10,10))
                break


        redrawWindow(win)


    pass



main()
```