Plagiarism Detector

Date: 2020-04-07

PLAGIARISM SCAN REPORT

0%	100%	532	3414
Plagiarised	Unique	Words	Characters

Exclude Url : None

Content Checked For Plagiarism

import turtle import colorsys import math turtle.setup(1000,600) turtle.hideturtle() turtle.title("Tower of Hanoi") turtle.speed(1) turtle.tracer(0,0) n=5 peg_height=300 ring_height_max=10 ring_width_max=150 ring_width_min=30 ring_delta=15 ring_delta_max=30 ring_height=30 animation_step=2 A =[] # list of rings in peg A B =[] c =[] T =[] # list of turtles def draw_line(x,y,heading,length,pensize,color): turtle.up() turtle.goto(x,y) turtle.seth(heading) turtle.down() turtle.color(color) turtle.pensize(pensize) turtle.fd(length) def draw_scene(): turtle.bgcolor('black') draw_line(-600,-100,0,1200,10,'white') for i in range(-250,251,250): draw_line(i,-93,90,peg_height,5,'white') def initialize(): global ring_width_max,ring_width_min,ring_ratio,ring_delta for i in range(n): A.append(i) t = turtle.turtle() t.hideturtle() t.speed(0) t.pencolor('black') t.fillcolor('light blue') T.append(t) ring_delta = min(135/(n-1),ring_delta_max) def draw_single_ring(r,x,k,extra=0): global ring_delta w = ring_width_max - ring_dekta*(r-1) T[r].up() T[r].goto(x-w/2,-95+ring_height*k + extra) T[r].down() T[r].seth(0) T[r].begin_fill() for i in range(2): T[r].fd(w) T[r].left(90) T[r].fd(ring_height) T[r].left(90) T[r].end_fill() def draw_rings(): for i in range(len(A)): draw_single_ring(A[i],-250,i) for i in range(len(B)): draw_single_ring(B[i],0,i) for i in range(len(C)): draw_single_ring(C[i],250,i) def move_ring(PP,QQ): if PP == "A": x = -250 p = A elif PP == "B": x = 0 P = B else: x = 250 P = C if QQ =="A": x2 = -250 Q = A elif QQ == "B": x2 = 0 Q = B else: x2 = 250 Q = C for extra in range(1,250-(-95+ring_height*(len(P)-1)),animation_step): T[P[len(p)-1]].clear() draw_single_ring(P[len(P)-1],x,len(P)-1,extra) turtle.update() T[P[len(P)-1]].clear() draw_single_ring(P[len(P)-1],x,len(P)-1,extra) turtle.update() tp = x if x2 > x: step = animation_step else: step = -animation_step for tp in range(x,x2,step): T[P[len(p)-1]].clear() draw-single_ring(P[len(P)-1],tp,len(P)-1,extra) turtle.update() T[P[len(P)-1]].clear() draw_single_ring(P[len(P)-1],x2,len(P)-1,extra) turtle.update() Q.append(P[len(P)-1]) del P[-1] for extra in range(250-(-95+ring_height*(len(Q)-1)),0,-animation_step): T[Q[len(Q)-1]].clear() draw_single_ring(Q[len(Q)-1],len(Q)-1,extra) turtle.update() T[Q[len(Q)-1]].clear() draw_single_ring(Q[len(Q)-1],x2,len(Q)-1) turtle.update() return #move rings in x to z def tower_of_hanoi(X,Y,z,n): if n== 1: move_ring(X,Z) return tower_of_hanoi(X,Z,Y,n-1) move_ring(X,Z) tower_of_hanoi(Y,X,Z,n-1) draw_scene() turtle.update() n = int(turtle.numinput('Number of Rings','Please enter of Rings: ',5,2,10)) initialize() draw_rings() tower_of_hanoi("A","B","C",n) turtle.update()

Plagiarism Detector