# Weather Forecast App

from tkinter import \*

from configparser import ConfigParser

from tkinter import messagebox

import requests

import json

url= "http://api.openweathermap.org/data/2.5/weather?q={}&appid={}"

config\_file='config.ini'

config= ConfigParser()

config.read(config\_file)

api\_key=config['api\_key']['key']

def get\_weather(city):

result=requests.get(url.format(city,api\_key))

if result:

json=result.json() #City,country, celsius, fahrenheit, icon, weather

city= json['name']

country=json['sys']['country']

kelvin\_temp=json['main']['temp']

celsius\_temp= kelvin\_temp-273.15

fahr\_temp= (kelvin\_temp-273.15) \* 9/5 + 32

icon=json['weather'][0]['icon']

weather=json['weather'][0]['main']

final=(city,country,celsius\_temp,fahr\_temp,icon,weather)

return final

else:

return None

def search():

city=city\_text.get()

weather=get\_weather(city)

if weather:

location\_lab['text']= "{}, {}".format(weather[0], weather[1])

img["file"]="Weather\_icons/{}.png".format(weather[4])

temp\_lab['text']="{:.2f}°C, {:.2f}°F".format(weather[2],weather[3])

weather\_lab['text']= weather[5]

else:

messagebox.showerror('Error',"City NOT FOUND {}".format(city))

app=Tk()

app.title("Weather app")

app.geometry("700x350")

city\_text=StringVar()

city\_entry= Entry(app, textvariable=city\_text,width=12,font=(15))

city\_entry.pack()

search\_but=Button(app,text="Search",width=12,command=search,font=('bold',18))

search\_but.pack()

location\_lab= Label(app,text="", font=('bold',18))

location\_lab.pack() #display the location on the screen

img= PhotoImage(file='')

image=Label(app,image=img)

image.pack()

temp\_lab=Label(app,text="",font=("bold",15))

temp\_lab.pack() #display the Temperature on the screen

weather\_lab=Label(app,text="",font=("bold",15))

weather\_lab.pack() #display the Weather on the screen

app.mainloop()

##### SHAPES USING TURTLE

# Spiral Flower

import turtle

tina = turtle.Turtle()

tina.speed(10)

tina.hideturtle()

background = turtle.Screen()

tina\_background =input("What color should the background be? ") #User defining the BG COLOUR

tina\_color =input("What color should Tina use to draw? ") #Pen Colour

background.bgcolor(tina\_background)

tina.color(tina\_color) #Co-ordinates for the spirals

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

tina.left(45)

tina.forward(20)

tina.circle(40)

#Extra

#The process will recur if the user wants to draw another layer

tina.left(46)

tina.forward(20)

tina.circle(40)

tina.left(46)

tina.forward(20)

tina.circle(40)

tina.left(46)

tina.forward(20)

tina.circle(40)

tina.left(46)

tina.forward(20)

tina.circle(40)

tina.left(46)

tina.forward(20)

tina.circle(40)

tina.left(46)

tina.forward(20)

tina.circle(40)

tina.left(46)

tina.forward(20)

tina.circle(40)

tina.left(46)

tina.forward(20)

tina.circle(40)

##### SPIRAL FLOWER with MANY LAYERS

import turtle

import math

import random

wn = turtle.Screen()

wn.bgcolor('black')

A = turtle.Turtle()

A.speed(0)

A.color('white')

rotate=int(360)

def drawCircles(t,size):

for i in range(10):

t.circle(size)

size=size-4

def drawSpecial(t,size,repeat):

for i in range (repeat):

drawCircles(t,size)

t.right(360/repeat)

drawSpecial(A,100,10)

S = turtle.Turtle()

S.speed(0)

S.color('yellow')

rotate=int(90)

def drawCircles(t,size):

for i in range(4):

t.circle(size)

size=size-10

def drawSpecial(t,size,repeat):

for i in range (repeat):

drawCircles(t,size)

t.right(360/repeat)

drawSpecial(S,100,10)

David = turtle.Turtle()

David.speed(0)

David.color('blue')

rotate=int(80)

def drawCircles(t,size):

for i in range(4):

t.circle(size)

size=size-5

def drawSpecial(t,size,repeat):

for i in range (repeat):

drawCircles(t,size)

t.right(360/repeat)

drawSpecial(David,100,10)

RK = turtle.Turtle()

RK.speed(0)

RK.color('orange')

rotate=int(90)

def drawCircles(t,size):

for i in range(4):

t.circle(size)

size=size-19

def drawSpecial(t,size,repeat):

for i in range (repeat):

drawCircles(t,size)

t.right(360/repeat)

drawSpecial(RK,100,10)

W = turtle.Turtle()

W.speed(0)

W.color('pink')

rotate=int(90)

def drawCircles(t,size):

for i in range(4):

t.circle(size)

size=size-20

def drawSpecial(t,size,repeat):

for i in range (repeat):

drawCircles(t,size)

t.right(360/repeat)

drawSpecial(W,100,10)

####### Bat LOGO

import turtle

import math

myPen = turtle.Turtle()

myPen.speed(500)

window = turtle.Screen()

window.bgcolor("#000000") #defining the window

myPen.color("yellow")

zoom=20

myPen.left(90)

myPen.penup()

myPen.goto(-7\*zoom,0)

myPen.pendown()

for xz in range(-7\*zoom,-3\*zoom,1):

x=xz/zoom

absx=math.fabs(x)

y=1.5\*math.sqrt((-math.fabs(absx-1))\*math.fabs(3-absx)/((absx-1)\*(3-absx)))\*(1+math.fabs(absx-3)/(absx-3))\*math.sqrt(1-(x/7)\*\*2)+(4.5+0.75\*(math.fabs(x-0.5)+math.fabs(x+0.5))-2.75\*(math.fabs(x-0.75)+math.fabs(x+0.75)))\*(1+math.fabs(1-absx)/(1-absx))

myPen.goto(xz,y\*zoom)

for xz in range(-3\*zoom,-1\*zoom-1,1):

x=xz/zoom

absx=math.fabs(x)

y=(2.71052+1.5-0.5\*absx-1.35526\*math.sqrt(4-(absx-1)\*\*2))\*math.sqrt(math.fabs(absx-1)/(absx-1))

myPen.goto(xz,y\*zoom)

myPen.goto(-1\*zoom,3\*zoom)

myPen.goto(int(-0.5\*zoom),int(2.2\*zoom))

myPen.goto(int(0.5\*zoom),int(2.2\*zoom))

myPen.goto(1\*zoom,3\*zoom)

for xz in range(1\*zoom+1,3\*zoom+1,1):

x=xz/zoom

absx=math.fabs(x)

y=(2.71052+1.5-0.5\*absx-1.35526\*math.sqrt(4-(absx-1)\*\*2))\*math.sqrt(math.fabs(absx-1)/(absx-1))

myPen.goto(xz,y\*zoom)

for xz in range(3\*zoom+1,7\*zoom+1,1):

x=xz/zoom

absx=math.fabs(x)

y = 1.5\*math.sqrt((-math.fabs(absx-1))\*math.fabs(3-absx)/((absx-1)\*(3-absx)))\*(1+math.fabs(absx-3)/(absx-3))\*math.sqrt(1-(x/7)\*\*2)+(4.5+0.75\*(math.fabs(x-0.5)+math.fabs(x+0.5))-2.75\*(math.fabs(x-0.75)+math.fabs(x+0.75)))\*(1+math.fabs(1-absx)/(1-absx))

myPen.goto(xz,y\*zoom)

for xz in range(7\*zoom,4\*zoom,-1):

x=xz/zoom

absx=math.fabs(x)

y=(-3)\*math.sqrt(1-(x/7)\*\*2) \* math.sqrt(math.fabs(absx-4)/(absx-4))

myPen.goto(xz,y\*zoom)

for xz in range(4\*zoom,-4\*zoom,-1):

x=xz/zoom

absx=math.fabs(x)

y=math.fabs(x/2)-0.0913722\*x\*\*2-3+math.sqrt(1-(math.fabs(absx-2)-1)\*\*2)

myPen.goto(xz,y\*zoom)

for xz in range(-4\*zoom-1,-7\*zoom-1,-1):

x=xz/zoom

absx=math.fabs(x)

y =(-3)\*math.sqrt(1-(x/7)\*\*2) \* math.sqrt(math.fabs(absx-4)/(absx-4))

myPen.goto(xz,y\*zoom)

myPen.penup()

myPen.goto(300,300)

####Iron Man

from turtle import Turtle, Screen

from time import sleep

#Top

piece1 = [ [0, -2, -3.5, -6.5, -8.5, -8.5, -8.0, -8.5, -7.5, -7.0, -2.0, 0],[0, 0, 7, 5.5, 4, -1, -4, -5.5, -6.5, -5.5, -7.0, -7.0] ]

###Middle piece

piece2 = [ [0, -2.0, -2.5, -5.0, -6.5, -8.8, -9.3, -9.3, -6.0, -5.5, -4.0, -3.2, 0], [-7.5, -7.5, -8.0, -8.3, -8.0, -6.0, -7.5, -8.0, -14.5, -16.5, -17.5, -16.5, -16.5] ]

###Bottom

piece3 = [ [0, -3.0, -4.0, -5.5, -6.0, -4.5, -3.0, -1.5, -1.0, 0], [-17.0, -17.0, -18.0, -17.0, -18.5, -20.0, -19.0, -19.0, -18.5, -18.5] ]

t=Turtle()

s=Screen()

t.hideturtle()

s.bgcolor("#ba161e") #Dark red

s.setup(500, 600)

x\_offset, y\_offset = 0, 120

zoom\_factor = 15

t.speed(2)

def draw\_piece(piece):

t.penup()

t.goto(piece[0][0] \* zoom\_factor + x\_offset, piece[1][0] \* zoom\_factor + y\_offset)

t.pendown()

t.color('#fab104') #Light Yellow

t.begin\_fill()

for i in range(1, len(piece[0])):

x, y = piece[0][i] \* zoom\_factor + x\_offset, piece[1][i] \* zoom\_factor + y\_offset

t.goto(x, y)

#color("#f19100")

for i in range(len(piece[0])-1, -1, -1):

x, y = piece[0][i] \* zoom\_factor \* -1 + x\_offset, piece[1][i] \* zoom\_factor + y\_offset

t.goto(x, y)

t.end\_fill()

draw\_piece(piece1)

draw\_piece(piece2)

draw\_piece(piece3)

sleep(4)