**1)**

**import** tkinter  
*# for creating random numbers.***import** random  
  
*# the list of possible colour.*colours = [**'Red'**, **'Blue'**, **'Green'**, **'Pink'**, **'Black'**, **'Yellow'**, **'Orange'**, **'White'**, **'Purple'**, **'Brown'**]  
*# the player's score, initially 0.*score = 0  
*# the game time left, initially 30 seconds.*timeleft = 30  
  
  
*# a function that will start the game.***def** startGame(event):  
 *# if there's still time left...* **if** timeleft == 30:  
 *# start the countdown timer.* countdown()  
  
 *# run the function to choose the next colour.* nextColour()  
  
  
*# function to choose and display the next colour.***def** nextColour():  
 *# use the globally declared 'score' and 'play' variables above.* **global** score  
 **global** timeleft  
  
 *# if a game is currently in play...* **if** timeleft > 0:  
  
 *# ...make the text entry box active.* e.focus\_set()  
  
 *# if the colour typed is equal to the colour of the text...* **if** e.get().lower() == colours[1].lower():  
 *# ...add one to the score.* score += 1  
  
 *# clear the text entry box.* e.delete(0, tkinter.END)  
 *# shuffle the list of colours.* random.shuffle(colours)  
 *# change the colour to type, by changing the text \_and\_ the colour to a random colour value* label.config(fg=str(colours[1]), text=str(colours[0]))  
 *# update the score.* scoreLabel.config(text=**"Score: "** + str(score))  
  
  
*# a countdown timer function.***def** countdown():  
 *# use the globally declared 'play' variable above.* **global** timeleft  
  
 *# if a game is in play...* **if** timeleft > 0:  
 *# decrement the timer.* timeleft -= 1  
 *# update the time left label.* timeLabel.config(text=**"Time left: "** + str(timeleft))  
 *# run the function again after 1 second.* timeLabel.after(1000, countdown)  
  
  
*# create a GUI window.*root = tkinter.Tk()  
*# set the title.*root.title(**"TTCANTW"**)  
*# set the size.*root.geometry(**"375x200"**)  
  
*# add an instructions label.*instructions = tkinter.Label(root, text=**"Type in the colour of the words, and not the word text!"**,  
 font=(**'Helvetica'**, 12))  
instructions.pack()  
  
*# add a score label.*scoreLabel = tkinter.Label(root, text=**"Press enter to start"**, font=(**'Helvetica'**, 12))  
scoreLabel.pack()  
  
*# add a time left label.*timeLabel = tkinter.Label(root, text=**"Time left: "** + str(timeleft), font=(**'Helvetica'**, 12))  
timeLabel.pack()  
  
*# add a label for displaying the colours.*label = tkinter.Label(root, font=(**'Helvetica'**, 60))  
label.pack()  
  
*# add a text entry box for typing in colours.*e = tkinter.Entry(root)  
*# run the 'startGame' function when the enter key is pressed.*root.bind(**'<Return>'**, startGame)  
e.pack()  
*# set focus on the entry box.*e.focus\_set()  
  
*# start the GUI*root.mainloop()

2)

**import** random  
**import** time  
**import** os  
  
**def** initGrid(cols, rows, array):  
 **for** i **in** range(rows):  
 arrayRow = []  
 **for** j **in** range(cols):  
 **if** (i == 0 **or** j == 0 **or** (i == rows - 1) **or** (j == cols - 1)):  
 arrayRow += [-1]  
 **else**:  
 ran = random.randint(0, 3)  
 **if** ran == 0:  
 arrayRow += [1]  
 **else**:  
 arrayRow += [0]  
 array += [arrayRow]  
  
  
**def** printGen(cols, rows, array, genNo):  
 os.system(**"cls"**)  
  
 print(**"Game of Life -- Generation "** + str(genNo + 1))  
  
 **for** i **in** range(rows):  
 **for** j **in** range(cols):  
 **if** array[i][j] == -1:  
 print(**"#"**, end=**" "**)  
 **elif** array[i][j] == 1:  
 print(**"."**, end=**" "**)  
 **else**:  
 print(**" "**, end=**" "**)  
 print(**"\n"**)  
  
  
  
**def** processNextGen(cols, rows, cur, nxt):  
 **for** i **in** range(1, rows - 1):  
 **for** j **in** range(1, cols - 1):  
 nxt[i][j] = processNeighbours(i, j, cur)  
  
  
  
**def** processNeighbours(x, y, array):  
 nCount = 0  
 **for** j **in** range(y - 1, y + 2):  
 **for** i **in** range(x - 1, x + 2):  
 **if not** (i == x **and** j == y):  
 **if** array[i][j] != -1:  
 nCount += array[i][j]  
 **if** array[x][y] == 1 **and** nCount < 2:  
 **return** 0  
 **if** array[x][y] == 1 **and** nCount > 3:  
 **return** 0  
 **if** array[x][y] == 0 **and** nCount == 3:  
 **return** 1  
 **else**:  
 **return** array[x][y]  
  
  
ROWS = 11  
COLS = 39  
GENERATIONS = 100  
DELAY = 0.2  
  
thisGen = []  
nextGen = []  
  
initGrid(COLS, ROWS, thisGen)  
initGrid(COLS, ROWS, nextGen)  
  
**for** gens **in** range(GENERATIONS):  
 printGen(COLS, ROWS, thisGen, gens)  
 processNextGen(COLS, ROWS, thisGen, nextGen)  
 time.sleep(DELAY)  
 thisGen, nextGen = nextGen, thisGen  
input(**"Finished. Press <return> to quit."**)