*# GUI.py*import pygame  
from Sudoku import solve, valid  
import time  
pygame.font.init()  
  
  
class Grid:  
 board = [  
 [7, 8, 0, 4, 0, 0, 1, 2, 0],  
 [6, 0, 0, 0, 7, 5, 0, 0, 9],  
 [0, 0, 0, 6, 0, 1, 0, 7, 8],  
 [0, 0, 7, 0, 4, 0, 2, 6, 0],  
 [0, 0, 1, 0, 5, 0, 9, 3, 0],  
 [9, 0, 4, 0, 6, 0, 0, 0, 5],  
 [0, 7, 0, 3, 0, 0, 0, 1, 2],  
 [1, 2, 0, 0, 0, 7, 4, 0, 0],  
 [0, 4, 9, 2, 0, 6, 0, 0, 7]  
 ]  
  
 def \_\_init\_\_(self, rows, cols, width, height):  
 self.rows = rows  
 self.cols = cols  
 self.cubes = [[Cube(self.board[i][j], i, j, width, height) for j in range(cols)] for i in range(rows)]  
 self.width = width  
 self.height = height  
 self.model = None  
 self.selected = None  
  
 def update\_model(self):  
 self.model = [[self.cubes[i][j].value for j in range(self.cols)] for i in range(self.rows)]  
  
 def place(self, val):  
 row, col = self.selected  
 if self.cubes[row][col].value == 0:  
 self.cubes[row][col].set(val)  
 self.update\_model()  
  
 if valid(self.model, val, (row,col)) and solve(self.model):  
 return True  
 else:  
 self.cubes[row][col].set(0)  
 self.cubes[row][col].set\_temp(0)  
 self.update\_model()  
 return False  
  
 def sketch(self, val):  
 row, col = self.selected  
 self.cubes[row][col].set\_temp(val)  
  
 def draw(self, win):  
 *# Draw Grid Lines* gap = self.width / 9  
 for i in range(self.rows+1):  
 if i % 3 == 0 and i != 0:  
 thick = 4  
 else:  
 thick = 1  
 pygame.draw.line(win, (0,0,0), (0, i\*gap), (self.width, i\*gap), thick)  
 pygame.draw.line(win, (0, 0, 0), (i \* gap, 0), (i \* gap, self.height), thick)  
  
 *# Draw Cubes* for i in range(self.rows):  
 for j in range(self.cols):  
 self.cubes[i][j].draw(win)  
  
 def select(self, row, col):  
 *# Reset all other* for i in range(self.rows):  
 for j in range(self.cols):  
 self.cubes[i][j].selected = False  
  
 self.cubes[row][col].selected = True  
 self.selected = (row, col)  
  
 def clear(self):  
 row, col = self.selected  
 if self.cubes[row][col].value == 0:  
 self.cubes[row][col].set\_temp(0)  
  
 def click(self, pos):  
 *"""  
 :param: pos  
 :return: (row, col)  
 """* if pos[0] < self.width and pos[1] < self.height:  
 gap = self.width / 9  
 x = pos[0] // gap  
 y = pos[1] // gap  
 return (int(y),int(x))  
 else:  
 return None  
  
 def is\_finished(self):  
 for i in range(self.rows):  
 for j in range(self.cols):  
 if self.cubes[i][j].value == 0:  
 return False  
 return True  
  
  
class Cube:  
 rows = 9  
 cols = 9  
  
 def \_\_init\_\_(self, value, row, col, width ,height):  
 self.value = value  
 self.temp = 0  
 self.row = row  
 self.col = col  
 self.width = width  
 self.height = height  
 self.selected = False  
  
 def draw(self, win):  
 fnt = pygame.font.SysFont(**"comicsans"**, 40)  
  
 gap = self.width / 9  
 x = self.col \* gap  
 y = self.row \* gap  
  
 if self.temp != 0 and self.value == 0:  
 text = fnt.render(str(self.temp), 1, (128,128,128))  
 win.blit(text, (x+5, y+5))  
 elif not(self.value == 0):  
 text = fnt.render(str(self.value), 1, (0, 0, 0))  
 win.blit(text, (x + (gap/2 - text.get\_width()/2), y + (gap/2 - text.get\_height()/2)))  
  
 if self.selected:  
 pygame.draw.rect(win, (255,0,0), (x,y, gap ,gap), 3)  
  
 def set(self, val):  
 self.value = val  
  
 def set\_temp(self, val):  
 self.temp = val  
  
  
def redraw\_window(win, board, time, strikes):  
 win.fill((255,255,255))  
 *# Draw time* fnt = pygame.font.SysFont(**"comicsans"**, 40)  
 text = fnt.render(**"Time: "** + format\_time(time), 1, (0,0,0))  
 win.blit(text, (540 - 160, 560))  
 *# Draw Strikes* text = fnt.render(**"X "** \* strikes, 1, (255, 0, 0))  
 win.blit(text, (20, 560))  
 *# Draw grid and board* board.draw(win)  
  
  
def format\_time(secs):  
 sec = secs%60  
 minute = secs//60  
 hour = minute//60  
  
 mat = **" "** + str(minute) + **":"** + str(sec)  
 return mat  
  
  
def main():  
 win = pygame.display.set\_mode((540,600))  
 pygame.display.set\_caption(**"Sudoku"**)  
 board = Grid(9, 9, 540, 540)  
 key = None  
 run = True  
 start = time.time()  
 strikes = 0  
 while run:  
  
 play\_time = round(time.time() - start)  
  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 run = False  
 if event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_1:  
 key = 1  
 if event.key == pygame.K\_2:  
 key = 2  
 if event.key == pygame.K\_3:  
 key = 3  
 if event.key == pygame.K\_4:  
 key = 4  
 if event.key == pygame.K\_5:  
 key = 5  
 if event.key == pygame.K\_6:  
 key = 6  
 if event.key == pygame.K\_7:  
 key = 7  
 if event.key == pygame.K\_8:  
 key = 8  
 if event.key == pygame.K\_9:  
 key = 9  
 if event.key == pygame.K\_DELETE:  
 board.clear()  
 key = None  
 if event.key == pygame.K\_RETURN:  
 i, j = board.selected  
 if board.cubes[i][j].temp != 0:  
 if board.place(board.cubes[i][j].temp):  
 print(**"Success"**)  
 else:  
 print(**"Wrong"**)  
 strikes += 1  
 key = None  
  
 if board.is\_finished():  
 print(**"Game over"**)  
 run = False  
  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 pos = pygame.mouse.get\_pos()  
 clicked = board.click(pos)  
 if clicked:  
 board.select(clicked[0], clicked[1])  
 key = None  
  
 if board.selected and key != None:  
 board.sketch(key)  
  
 redraw\_window(win, board, play\_time, strikes)  
 pygame.display.update()  
  
  
main()  
pygame.quit()