15IT323E GAME PROGRAMMING

A PROJECT REPORT

Submitted by

Dhir Patel [Reg No: RA1711020010077]

In submission for assessment and award

of Grade

BACHELOR OF TECHNOLOGY

in

SOFTWARE ENGINEERING

of

FACULTY OF ENGINEERING & TECHNOLOGY

S.R.M Nagar, Kattankulathur, Kancheepuram District

OCTOBER 2019

Table of Contents:-

S. No.	Title	Page
1	Introduction	3
2	Objective of Project	4
3	Explanation of Project	5
4	Conclusion	23

INTRODUCTION

Game development is the art of creating games. It has phases of design, development, and deployment of the game. It involves concept building, design, build, vigorous testing and release. While developing a game it is important to keep in mind the factors such as mechanics, rewards, engagement of players and level design.

The project undertaken in here is a driving and obstacle avoidance game. It is developed using pygame library in python language.

Objective of Project

Objective of the project is to develop a basic starter game for the graded assignment towards unit-5 of Game Programming subject.

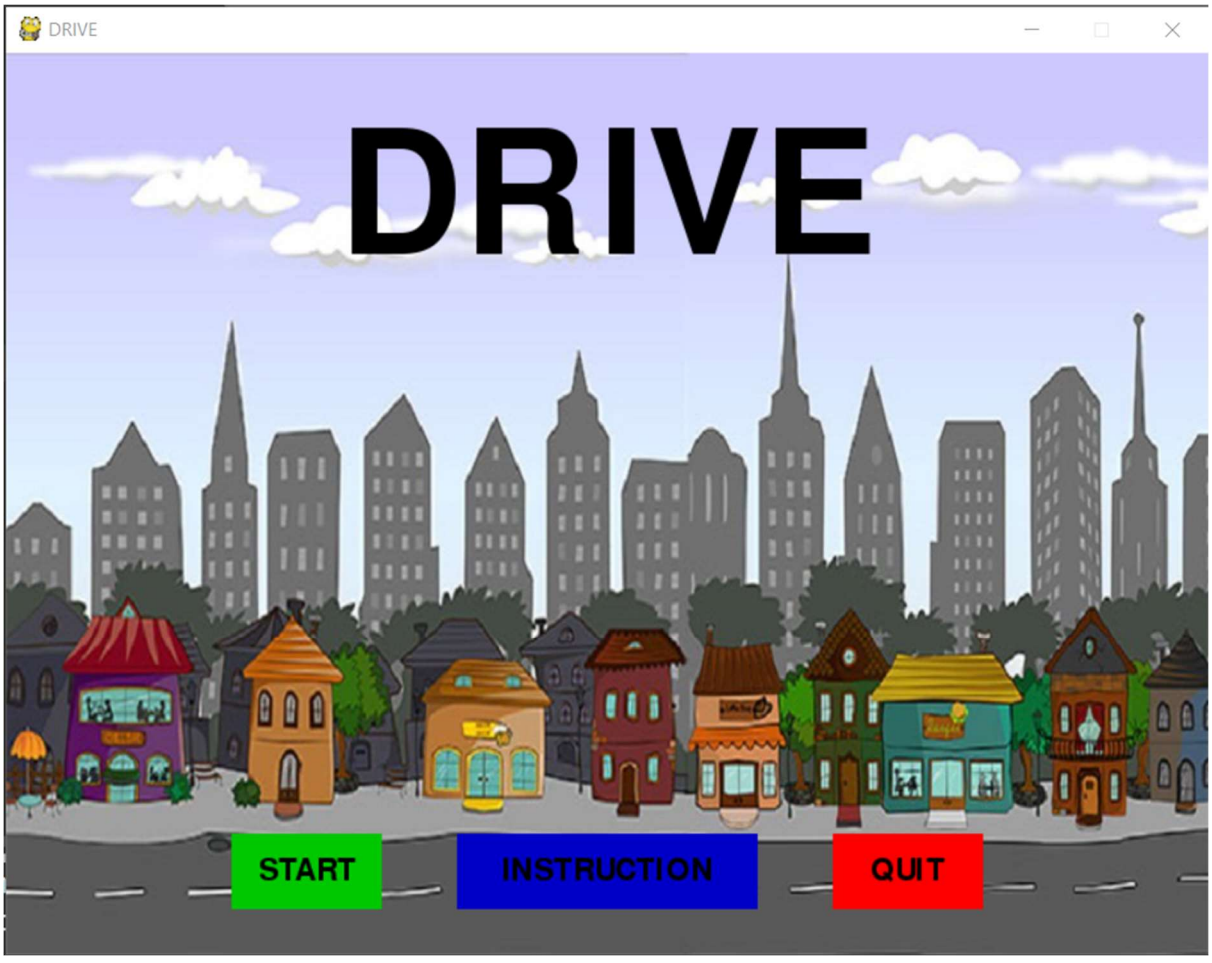
The game developed in here is called DRIVE. It is a never-ending level-oriented game. Its basic concept is obstacle avoidance. You have to just drive a car along the road and avoid other cars(obstacles), to play the game. The speed of the car increases steadily after every level. The game also keeps track of your score as the number of cars passed and a score object which multiplies the object passed with a fixed constant.

Explanation of Project

The game is developed in python using a game developing library called Pygame. The game is developed as a showcase of object avoidance. The game is called DRIVE. The basic structure is to drive a car and avoid other cars while overtaking and not to bump into them. It is a never-ending game and a level-oriented game. Where at each level the speed of the player's car increases by a fixed constant.

The game has two different windows: -

➔ The Main Screen: -

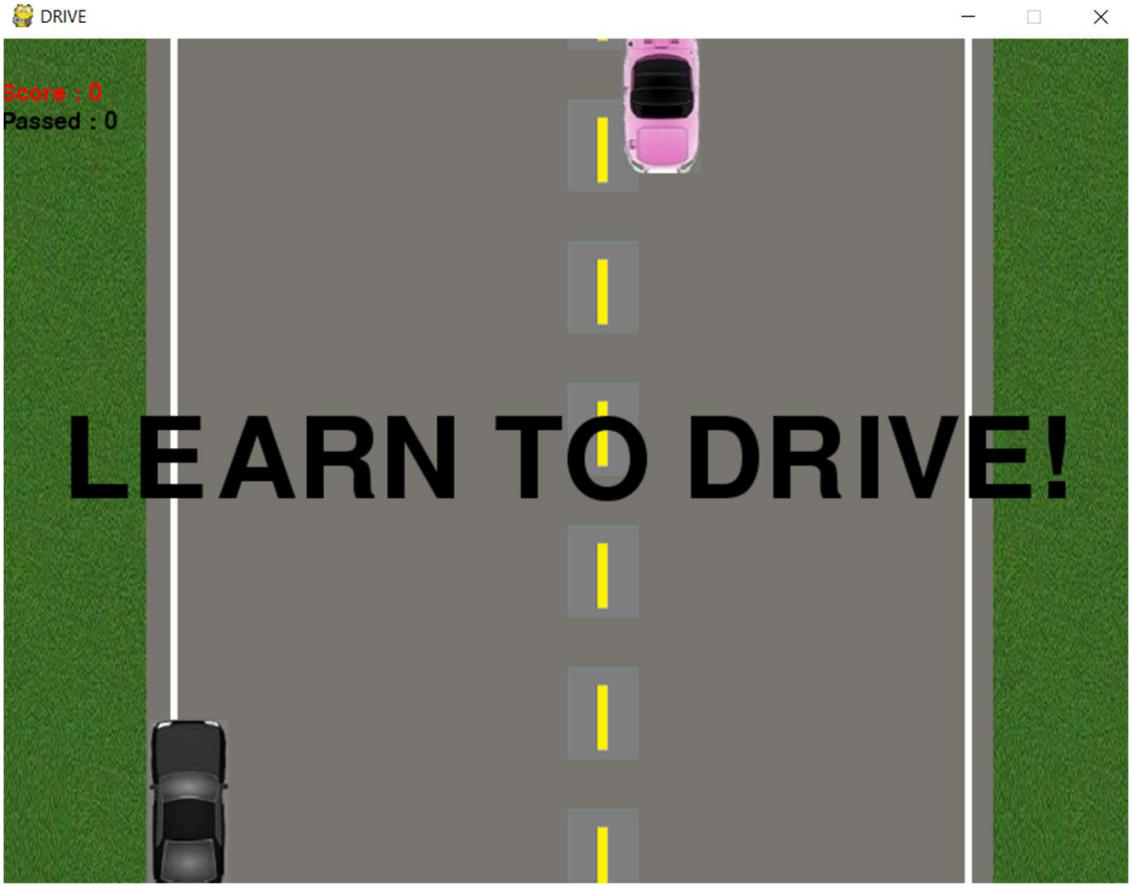


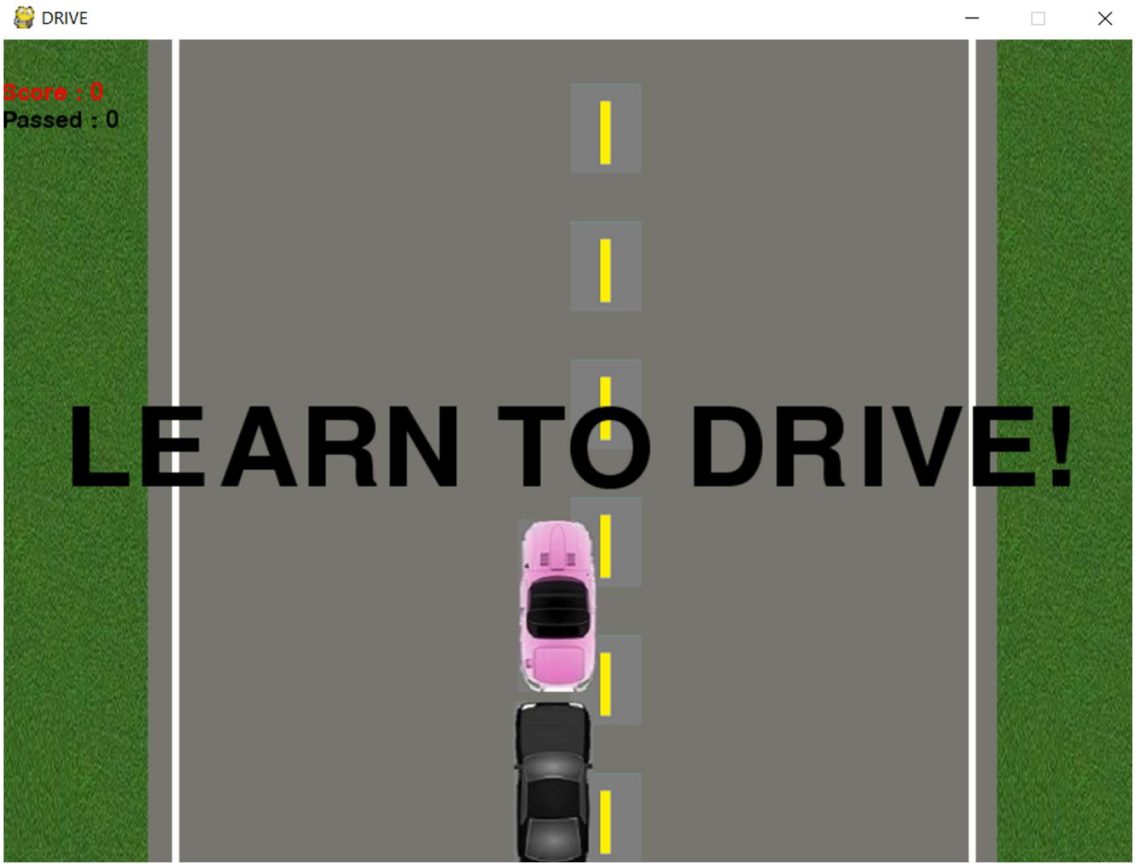
➔ The Game Screen: -

The game screen consists of different elements such as the green field of grass developed by putting strips of images along the both vertical sides. Then comes the white border which also works to check for the crash of the car at the road border. Then the road where the obstacle cars and players car are placed.

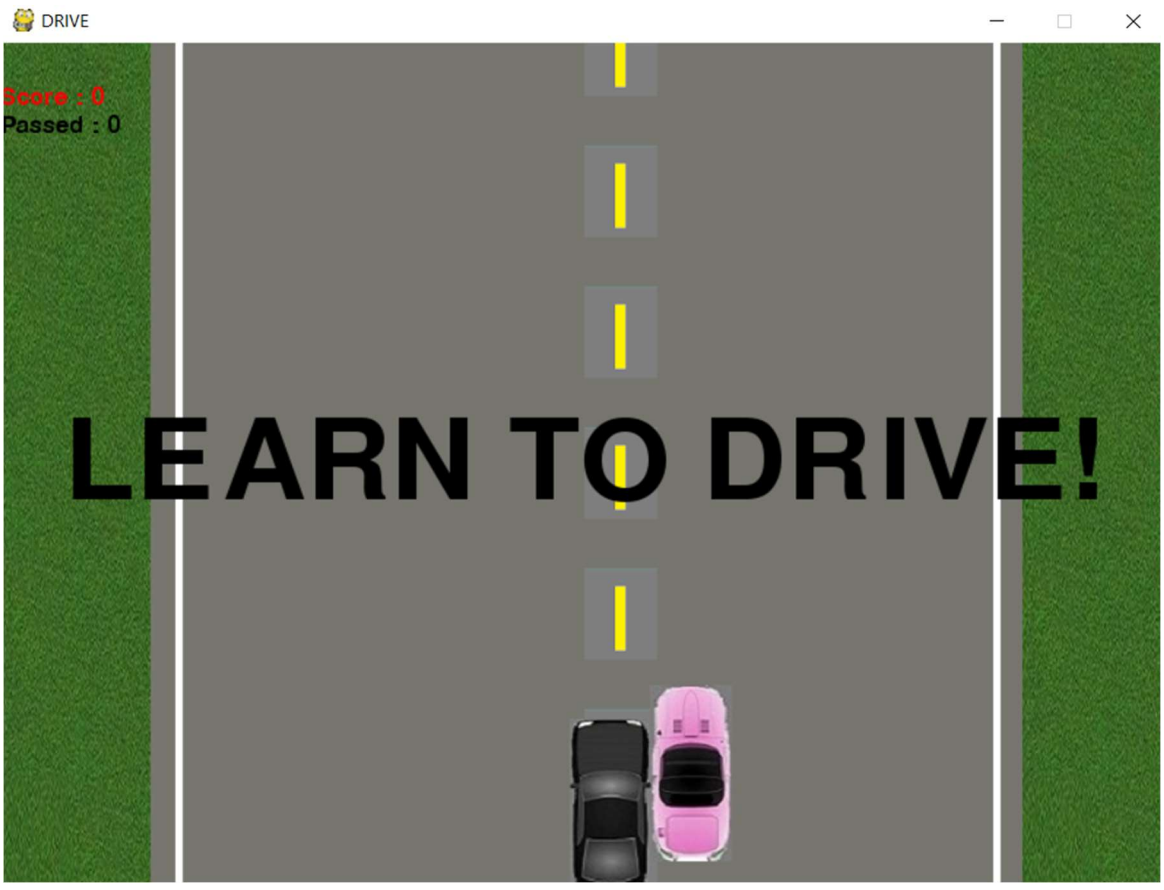
The game shows the game over message if the player's car touches the obstacle car or the road boundaries on the either sides.

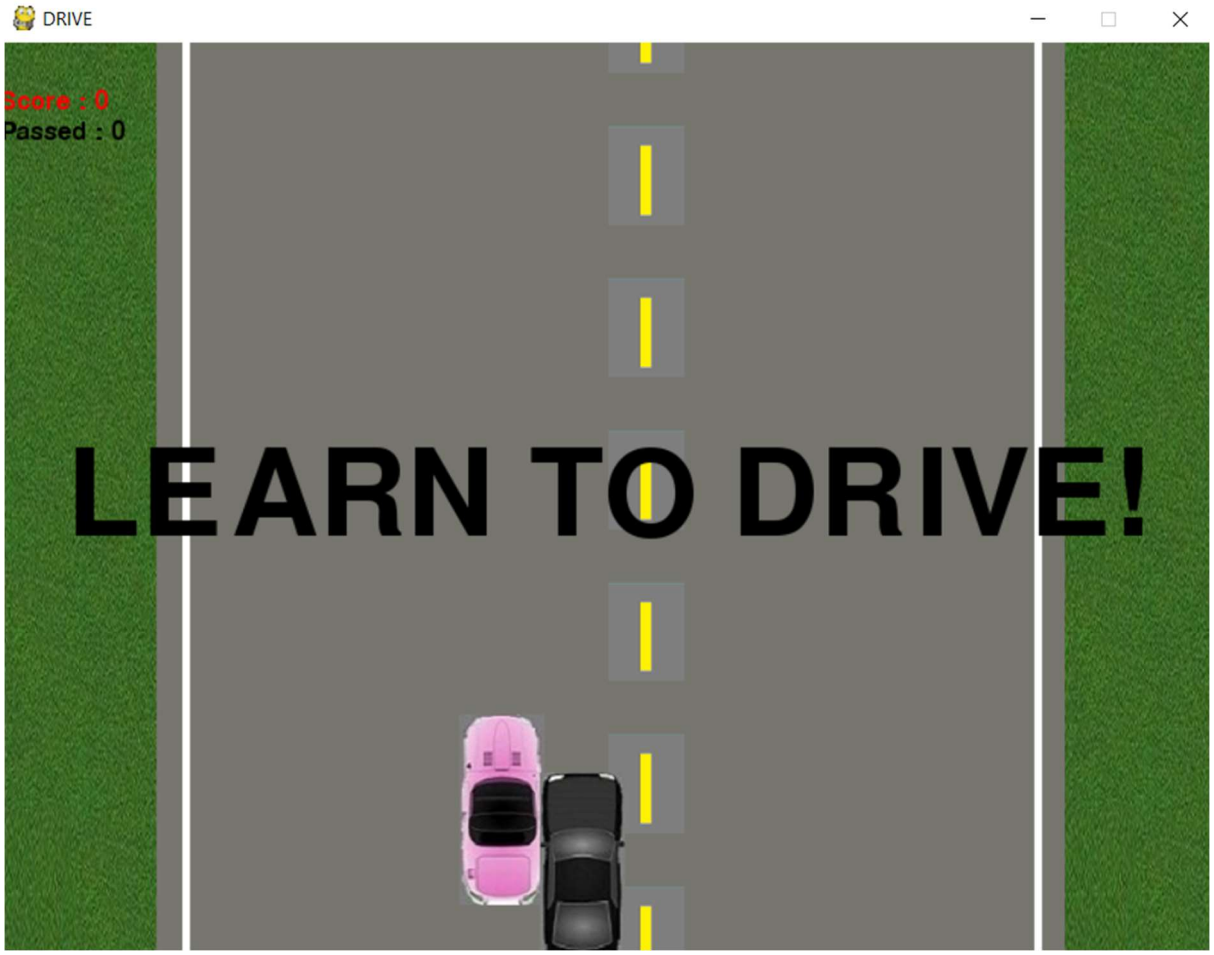
Some crash types images are given below: -






DRIVE
Score : 0
Passed : 0
LEARN TO DRIVE!


DRIVE
Score : 0
Passed : 0
LEARN TO DRIVE!



The crash in the games are identified by the pixel calculation of the obstacle car and the player's car. The crash over the sides are calculated by the check of boundary pixels defined.

As declared earlier that the game also has a level-system. In here the speed of the player's car increases with a constant factor with each passing level. Though the speed can be increased and decreased with their respective controls, the levels will increase the speed of the player's car with respect to the current speed. As the game being endless there is no limit to the levels. You can play as long you last with ever increasing speed.


DRIVE
Score : 90
Passed : 9
LEVEL 1


DRIVE
Score : 990
Passed : 99
LEVEL 10

The source code for the game is given below which was developed in python programming language.

➔ These lines of code are used for importing the required libraries.

```
1  # Headers
2  import pygame
3  import time
4  import random
```

➔ The following are the global color variables used to define the color used throughout the file.

```
9  # color == HexCode
10 gray=(119,118,110)
11 black=(0,0,0)
12 red=(255,0,0)
13 green=(0,200,0)
14 blue=(0,0,200)
15 bright_red=(255,0,0)
16 bright_green=(0,255,0)
17 bright_blue=(0,0,255)
```

➔ Initialization of pygame object for screen creation.

```
6  # initialization of the screen
7  pygame.init()
```

➔ Creating the screen definition.

```
# main screen sizes
display_width=800
display_height=600

# setting up display...
gamedisplays=pygame.display.set_mode((display_width,display_height))
pygame.display.set_caption("DRIVE")
clock=pygame.time.Clock()

```

➔ Loading all the required images.

```
# loading different images...
carimg=pygame.image.load('car1.jpg')
backgroundpic=pygame.image.load("download12.jpg")
yellow_strip=pygame.image.load("yellow strip.jpg")
strip=pygame.image.load("strip.jpg")
intro_background=pygame.image.load("background.jpg")
instruction_background=pygame.image.load("background2.jpg")
car_width=56
pause=False
```

➔ The recurring function to control the introduction loop.

```

# introduction screen...
def intro_loop():
    intro=True
    while intro:
        for event in pygame.event.get():
            if event.type==pygame.QUIT:
                pygame.quit()
                quit()
                sys.exit()
        gamedisplays.blit(intro_background,(0,0))
        largetext=pygame.font.Font('freesansbold.ttf',115)
        TextSurf,TextRect=text_objects("DRIVE",largetext)
        TextRect.center=(400,100)
        gamedisplays.blit(TextSurf,TextRect)
        button("START",150,520,100,50,green,bright_green,"play")
        button("QUIT",550,520,100,50,red,bright_red,"quit")
        button("INSTRUCTION",300,520,200,50,blue,bright_blue,"intro")
        pygame.display.update()
        clock.tick(50)
```

➔ The code to place all the necessary buttons on the introduction screen.

```
# buttons on main screen...
def button(msg,x,y,w,h,ic,ac,action=None):
    mouse=pygame.mouse.get_pos()
    click=pygame.mouse.get_pressed()
    if x+w>mouse[0]>x and y+h>mouse[1]>y:
        pygame.draw.rect(gamedisplays,ac,(x,y,w,h))
        if click[0]==1 and action!=None:
            if action=="play":
                countdown()
            elif action=="quit":
                pygame.quit()
                quit()
                sys.exit()
            elif action=="intro":
                introduction()
            elif action=="menu":
                intro_loop()
            elif action=="pause":
                paused()
            elif action=="unpause":
                unpaused()


    else:
        pygame.draw.rect(gamedisplays,ic,(x,y,w,h))
    smalltext=pygame.font.Font("freesansbold.ttf",20)
    textsurf,textrect=text_objects(msg,smalltext)
    textrect.center=((x+(w/2)),(y+(h/2)))
    gamedisplays.blit(textsurf,textrect)
```

➔ The instruction screen which can be accessed through the introduction screen.

```
87
88  # instructions screen...
89  def introduction():
90      introduction=True
91      while introduction:
92          for event in pygame.event.get():
93              if event.type==pygame.QUIT:
94                  pygame.quit()
95                  quit()
96                  sys.exit()
97          gamedisplays.blit(instruction_background,(0,0))
98          largetext=pygame.font.Font('freesansbold.ttf',80)
99          smalltext=pygame.font.Font('freesansbold.ttf',20)
100         mediumtext=pygame.font.Font('freesansbold.ttf',40)
```

```
100         mediumtext=pygame.font.Font('freesansbold.ttf',40)
101         textSurf,textRect=text_objects("This is a Driving game in which you dodge the cars to score!",smalltext)
102         textRect.center=((350),(200))
103         TextSurf,TextRect=text_objects("INSTRUCTION",largetext)
104         TextRect.center=((400),(100))
105         gamedisplays.blit(TextSurf,TextRect)
106         gamedisplays.blit(textSurf,textRect)
107         stextSurf,stextRect=text_objects("ARROW LEFT : MOVE LEFT",smalltext)
108         stextRect.center=((150),(400))
109         hTextSurf,hTextRect=text_objects("ARROW RIGHT : MOVE RIGHT" ,smalltext)
110         hTextRect.center=((150),(450))
111         atextSurf,atextRect=text_objects("A : ACCELERATOR",smalltext)
112         atextRect.center=((150),(500))
113         rtextSurf,rtextRect=text_objects("B : BRAKE ",smalltext)
114         rtextRect.center=((150),(550))
115         ptextSurf,ptextRect=text_objects("P : PAUSE  ",smalltext)
116         ptextRect.center=((150),(350))
117         sTextSurf,sTextRect=text_objects("CONTROLS",mediumtext)
118         sTextRect.center=((350),(300))
119         gamedisplays.blit(sTextSurf,sTextRect)
120         gamedisplays.blit(stextSurf,stextRect)
121         gamedisplays.blit(hTextSurf,hTextRect)
122         gamedisplays.blit(atextSurf,atextRect)
123         gamedisplays.blit(rtextSurf,rtextRect)
124         gamedisplays.blit(ptextSurf,ptextRect)
125         button("BACK",600,450,100,50,blue,bright_blue,"menu")
126         pygame.display.update()
127         clock.tick(30)
```

➔ The pause and resume features.

```
# pause functionality...
def paused():
    global pause

    while pause:
            for event in pygame.event.get():
                if event.type==pygame.QUIT:
                    pygame.quit()
                    quit()
                    sys.exit()
            gamedisplays.blit(instruction_background,(0,0))
            largetext=pygame.font.Font('freesansbold.ttf',115)
            TextSurf,TextRect=text_objects("PAUSED",largetext)
            TextRect.center=((display_width/2),(display_height/2))
            gamedisplays.blit(TextSurf,TextRect)
            button("CONTINUE",150,450,150,50,green,bright_green,"unpause")
            button("RESTART",350,450,150,50,blue,bright_blue,"play")
            button("MAIN MENU",550,450,200,50,red,bright_red,"menu")
            pygame.display.update()
            clock.tick(30)

def unpaused():
    global pause
    pause=False

```

➔ Creating the background for the countdown at the start of the game.

```
156 def countdown_background():
157     font=pygame.font.SysFont(None,25)
158     x=(display_width*0.45)
159     y=(display_height*0.8)
160     gamedisplays.blit(backgroundpic,(0,0))
161     gamedisplays.blit(backgroundpic,(0,200))
162     gamedisplays.blit(backgroundpic,(0,400))
163     gamedisplays.blit(backgroundpic,(700,0))
164     gamedisplays.blit(backgroundpic,(700,200))
165     gamedisplays.blit(backgroundpic,(700,400))
166     gamedisplays.blit(yellow_strip,(400,100))
167     gamedisplays.blit(yellow_strip,(400,200))
168     gamedisplays.blit(yellow_strip,(400,300))
169     gamedisplays.blit(yellow_strip,(400,400))
170     gamedisplays.blit(yellow_strip,(400,100))
171     gamedisplays.blit(yellow_strip,(400,500))
172     gamedisplays.blit(yellow_strip,(400,0))
173     gamedisplays.blit(yellow_strip,(400,600))
174     gamedisplays.blit(strip,(120,200))
175     gamedisplays.blit(strip,(120,0))
176     gamedisplays.blit(strip,(120,100))
177     gamedisplays.blit(strip,(680,100))
178     gamedisplays.blit(strip,(680,0))
179     gamedisplays.blit(strip,(680,200))
180     gamedisplays.blit(carimg,(x,y))
181     text=font.render("DODGED: 0",True, black)
182     score=font.render("SCORE: 0",True,red)
183     gamedisplays.blit(text,(0,50))
184     gamedisplays.blit(score,(0,30))
185     button("PAUSE",650,0,150,50,blue,bright blue,"pause")
```

➔ Creating the countdown.

```
def countdown():
    countdown=True

    while countdown:
            for event in pygame.event.get():
                if event.type==pygame.QUIT:
                    pygame.quit()
                    quit()
                    sys.exit()
            gamedisplays.fill(gray)
            countdown_background()
            largetext=pygame.font.Font('freesansbold.ttf',115)
            TextSurf,TextRect=text_objects("3",largetext)
            TextRect.center=((display_width/2),(display_height/2))
            gamedisplays.blit(TextSurf,TextRect)
            pygame.display.update()
            clock.tick(1)
            gamedisplays.fill(gray)
            countdown_background()
            largetext=pygame.font.Font('freesansbold.ttf',115)
            TextSurf,TextRect=text_objects("2",largetext)
            TextRect.center=((display_width/2),(display_height/2))
            gamedisplays.blit(TextSurf,TextRect)
            pygame.display.update()
            clock.tick(1)
```

```
            gamedisplays.fill(gray)
            countdown_background()
            largetext=pygame.font.Font('freesansbold.ttf',115)
            TextSurf,TextRect=text_objects("1",largetext)
            TextRect.center=((display_width/2),(display_height/2))
            gamedisplays.blit(TextSurf,TextRect)
            pygame.display.update()
            clock.tick(1)
            gamedisplays.fill(gray)
            countdown_background()
            largetext=pygame.font.Font('freesansbold.ttf',115)
            TextSurf,TextRect=text_objects("DRIVE!!!",largetext)
            TextRect.center=((display_width/2),(display_height/2))
            gamedisplays.blit(TextSurf,TextRect)
            pygame.display.update()
            clock.tick(1)
            game_loop()

```

➔ Code to choose a random car object.

```
# Different object display according to the random int passed in the function.
def obstacle(obs_startx,obs_starty,obs):
    if obs==0:
        obs_pic=pygame.image.load("car.jpg")
    elif obs==1:
        obs_pic=pygame.image.load("car1.jpg")
    elif obs==2:
        obs_pic=pygame.image.load("car2.jpg")
    elif obs==3:
        obs_pic=pygame.image.load("car4.jpg")
    elif obs==4:
        obs_pic=pygame.image.load("car5.jpg")
    elif obs==5:
        obs_pic=pygame.image.load("car6.jpg")
    elif obs==6:
        obs_pic=pygame.image.load("car7.jpg")
    gamedisplays.blit(obs_pic,(obs_startx,obs_starty))
```

➔ Scoring System.

```
# Score Board...
def score_system(passed,score):
    font=pygame.font.SysFont(None,25)
    text=font.render("Passed : "+str(passed),True,black)
    score=font.render("Score : "+str(score),True,red)
    gamedisplays.blit(text,(0,50))
    gamedisplays.blit(score,(0,30))
```

➔ Crash or game control function.

```
256  # Text formatting...
257  def text_objects(text,font):
258      textsurface=font.render(text,True,black)
259      return textsurface,textsurface.get_rect()
260
261  # Message Display
262  def message_display(text):
263      largetext=pygame.font.Font("freesansbold.ttf",80)
264      textsurf,textrect=text_objects(text,largetext)
265      textrect.center=((display_width/2),(display_height/2))
266      gamedisplays.blit(textsurf,textrect)
267      pygame.display.update()
268      time.sleep(3)
269      game_loop()
270
271  # calls the message display function with a Game over text...
272  def crash():
273      message_display("LEARN TO DRIVE!")
```

➔ Game background and player's car deployment.

```
275  # setting the gaming background...
276  def background():
277      gamedisplays.blit(backgroundpic,(0,0))
278      gamedisplays.blit(backgroundpic,(0,200))
279      gamedisplays.blit(backgroundpic,(0,400))
280      gamedisplays.blit(backgroundpic,(700,0))
281      gamedisplays.blit(backgroundpic,(700,200))
282      gamedisplays.blit(backgroundpic,(700,400))
283      gamedisplays.blit(yellow_strip,(400,0))
284      gamedisplays.blit(yellow_strip,(400,100))
285      gamedisplays.blit(yellow_strip,(400,200))
286      gamedisplays.blit(yellow_strip,(400,300))
287      gamedisplays.blit(yellow_strip,(400,400))
288      gamedisplays.blit(yellow_strip,(400,500))
289      gamedisplays.blit(strip,(120,0))
290      gamedisplays.blit(strip,(120,100))
291      gamedisplays.blit(strip,(120,200))
292      gamedisplays.blit(strip,(680,0))
293      gamedisplays.blit(strip,(680,100))
294      gamedisplays.blit(strip,(680,200))
295
296  # setting the origin for the player's car...
297  def car(x,y):
298      gamedisplays.blit(carimg,(x,y))
```

➔ The final function which controls every aspects of the game by calling the right functions defined above with the accordance of the user input event type.
It also declares the speed of the car and obstacles and the levels of the game.

```
300  # the control loop...
301  def game_loop():
302      global pause
303      # player's car coordinates...
304      x=(display_width*0.45)
305      y=(display_height*0.8)
306      x_change=0
307
308      # obstacle details...
309      obstacle_speed=9
310      obs=0
311      y_change=0
312      obs_startx=random.randrange(200,(display_width-200))
313      obs_starty=-750
314      obs_width=56
315      obs_height=125
316      passed=0
317      level=0
318      score=0
319      y2=7
320      fps=120
```

```
321
322     # event handler...
323     bumped=False
324     while not bumped:
325         for event in pygame.event.get():
326             if event.type==pygame.QUIT:
327                 pygame.quit()
328                 quit()
329 
330             if event.type==pygame.KEYDOWN:
331                 if event.key==pygame.K_LEFT:
332                     x_change=-5
333                 if event.key==pygame.K_RIGHT:
334                     x_change=5
335                 if event.key==pygame.K_a:
336                     obstacle_speed+=2
337                 if event.key==pygame.K_b:
338                     obstacle_speed-=2
339             if event.type==pygame.KEYUP:
340                 if event.key==pygame.K_LEFT or event.key==pygame.K_RIGHT:
341                     x_change=0
```

```
343         x+=x_change
344         pause=True
345         gamedisplays.fill(gray)
346 
347         rel_y=y2%backgroundpic.get_rect().width
348         gamedisplays.blit(backgroundpic,(0,rel_y-backgroundpic.get_rect().width))
349         gamedisplays.blit(backgroundpic,(700,rel_y-backgroundpic.get_rect().width))
350         if rel_y<800:
351             gamedisplays.blit(backgroundpic,(0,rel_y))
352             gamedisplays.blit(backgroundpic,(700,rel_y))
353             gamedisplays.blit(yellow_strip,(400,rel_y))
354             gamedisplays.blit(yellow_strip,(400,rel_y+100))
355             gamedisplays.blit(yellow_strip,(400,rel_y+200))
356             gamedisplays.blit(yellow_strip,(400,rel_y+300))
357             gamedisplays.blit(yellow_strip,(400,rel_y+400))
358             gamedisplays.blit(yellow_strip,(400,rel_y+500))
359             gamedisplays.blit(yellow_strip,(400,rel_y-100))
360             gamedisplays.blit(strip,(120,rel_y-200))
361             gamedisplays.blit(strip,(120,rel_y+20))
362             gamedisplays.blit(strip,(120,rel_y+30))
363             gamedisplays.blit(strip,(680,rel_y-100))
364             gamedisplays.blit(strip,(680,rel_y+20))
365             gamedisplays.blit(strip,(680,rel_y+30))
```

```
        # level and scoring system...
        obs_starty-=(obstacle_speed/4)
        obstacle(obs_startx,obs_starty,obs)
        obs_starty+=obstacle_speed
        car(x,y)
        score_system(passed,score)
        if x>690-car_width or x<110:
            crash()
        if x>display_width-(car_width+110) or x<110:
            crash()
        if obs_starty>display_height:
            obs_starty=0-obs_height
            obs_startx=random.randrange(170,(display_width-170))
            obs=random.randrange(0,7)
            passed=passed+1
            score=passed*10
            if int(passed)%10==0:
                level=level+1
                obstacle_speed+2
                largetext=pygame.font.Font("freesansbold.ttf",80)
                textsurf,textrect=text_objects("LEVEL "+str(level),largetext)
                textrect.center=((display_width/2),(display_height/2))
                gamedisplays.blit(textsurf,textrect)
                pygame.display.update()
                time.sleep(3)
```

```


if y<obs_starty+obs_height:
    if x > obs_startx and x < obs_startx + obs_width or x+car_width > obs_startx and x+car_width < obs_startx+obs_width:
        crash()
button("Pause",650,0,150,50,blue,bright_blue,"pause")
pygame.display.update()
clock.tick(60)
```

➔ Main function

```
# main Function...
if __name__ == "__main__":
    intro_loop()
    game_loop()
    pygame.quit()
    quit()
```

CONCLUSION

The game DRIVE was successfully developed and tested. The key concepts learned are Event management, design, creating objects, etc.