תיק פרוייקט – משחק בפייתון

“Gran Turismo”

מוגש על ידי איתי בר

24.5.2020

מגמת הנדסת תוכנה

בית ספר "ליאו באק"

מוגש לידי שרית לולב

תוכן עניינים

תיאור הפרוייקט

המשחק הוא משחק מכוניות פשוט בו על השחקן, בעזרת חצי המקלדת, לעבור בין שלושה נתיבים על מנת לעקוף את שאר המכוניות בכביש ולא להתנגש בהם. מקש חץ ימני יזיז את המכונית נתיב אחד ימינה ומקש חץ שמאלי יזיז את המכונית נתיב אחד שמאלה. צבירת הנקודות מתבצעת כל זמן לא הייתה התנגשות.

הוראות הפעלה

הוראות התקנה

על מנת להתקין את המשחק, קודם כל יש לוודא כי ipython 3 - pygame מותקנים על המחשב. במידה וpython מותקן על המחשב, ניתן להתקין את pygame על ידי כניסה לשורת הפקודה במחשב והכנסת השורה "pip install pygame". לאחר מכן, יש להוריד את קובץ ה-ZIP של המשחק למחשב ולבצע ייצוא. ניתן לעשות זאת על ידי לחיצה מקש ימני על הקובץ ובחירה באפשרות Extract All... ייפתח חלון בו יש להכניס או לבחור את המקום בו יותקן המשחק. לחיצה על Extract תתקין את המשחק.

הוראות הרצה

על מנת להריץ את המשחק את המשחק, יש לגשת אל התיקייה של המשחק, ללחוץ מקש ימני של העכבר על הקובץ "Gran_Turismo.py", לבחור באפשרות Open with, ולבחור בPython.

הוראות הפעלה

בפתיחת המשחק, ירד התפריט הראשי. על מנת להתחיל לשחק יש ללחוץ על הכפתור Play Now. לאחר מכן תחל ספירה של שלוש שניות שבסופן יחל המשחק. התזוזה בין הנתיבים ימינה שמאלה נעשית באמצעות מקש חץ ימני ומקש חץ שמאלי בהתאמה. ניתן להשהות את המשחק על ידי לחיצה על הכפתור בפינה השמאלית העליונה או לחיצה על מקש p במקלדת. יציאה מהשהייה תתבצע על ידי לחיצה על כל מקש במקלדת או בעכבר. המשחק יסתיים כאשר מכונית השחקן תתנגש באחת מכוניות המשחק ומיד ירד תפריט סיום המשחק. בתפריט סיום המשחק בצד שמאל ניתן יהיה לראות את טבלת השיאים שכוללת עד שמונה מקומות. בפינה הימנית העליונה של הטבלה ישנו לחצן אתחול, שלחיצה עליו תאתחל את טבלת השיאים. בפינה הימנית העליונה יהיה ניתן לראות את הניקוד הסופי במשחק האחרון. במידה והניקוד שהתקבל נכנס לטבלת השיאים, תהיה אפשרות להכניס שם עד שישה תווים. על מנת לשחק שוב, ניתן ללחוץ על כפתור Play Again או ללחוץ על מקש הרווח. על מנת לחזור לתפריט הראשי, יש ללחוץ על הכפתור Main Menu. ניתן לבצע השתקה של מוזיקת הרקע בכל זמן על ידי לחיצה על הכפתור המתאים בפינה השמאלית העליונה או לחיצה על מקש m במקלדת.

תיאור הפתרון

מבנה התוכנית הוא כך שבתחילה מסודרים הקבועים לשלושה קטגוריות: אלה שניתן לשנות בחופשיות, אלה שניתן לשנות אך לא מומלץ, ואלה שאסור לשנות. מתחת לקבועים מוגדרים כל הפונקציות של המשחק ואחר כך מאותחלים כל הsprites ונפתח חלון הpygame. מתחת, מוגדרת פונקציית הmain בה מאותחלים משתנים נוספים ונמצאת הלולאה הראשית של המשחק.

בתוך הלולאה הראשית של המשחק קיימות ארבע לולאות משנה: התפריט הראשי, תחילת המשחק, המשחק עצמו ותפריט סיום המשחק. בין התפריט הראשי לתחילת המשחק ולאחר תפריט סיום המשחק קיימת עוד לולאת משנה אשר נמצאת בפונקציה משלה אשר מטרתה לאפס את המסך ממכוניות בדרך אלגנטית.

העצים, הקווים והמכוניות שמורים כל אחד ברשימה אשר מאותחלת בתחילת הפונקציה הראשית ומתעדכנת במשך כל זמן הרצת התוכנית. כל מכונית למשל, שמורה כרשימה הכוללת את מיקומה הנוכחי, את סוג המכונית, את אורכה ואת רוחבה. רשימה זו של המכונית הבודדת שמורה ברשימה הכוללת של כל המכוניות. כאשר מתבצעת תנועה, הרשימות מתעדכנות כל פעם.

בעוד שהקווים נוצרים כל פעם באותו המקום ובסדר קבוע, העצים ושאר מכוניות המשחק נוצרים באופן אקראי אך מבוקר. כלומר, העצים נוצרים רק בצדדי המשחק ולא מפריעים לנראות הכביש והמכוניות לא מופיעות אחת על השנייה או מאפשרות מצב של חסימה בה לשחקן אין כל אפשרות לעבור בלי להתנגש. גם סוג המכונית נבחר באופן רנדומלי כל פעם.

גרפיקת התנועה של המכונית מתאפשרת משום ששאר הרקע של המשחק כמו העצים והפסים נעים אחורה במהירות קבועה בעוד שמכונית השחקן נשארת קבועה במקומה מה שיוצר אשליה שהמכונית נוסעת. גם שאר מכוניות הכביש נעות אחורה אך במהירות פחותה משום שגם להן יש מהירות מסוימת.

טבלת השיאים של המשחק מתאפשרת באמצעות שמירתה בקובץ טקסט נפרד אשר לא נמחק. אל קובץ זה נשמרים ונקראים השמות והנקודות. במידה והקובץ לא קיים, ייווצר חדש.

על מנת לבדוק אם הייתה התנגשות בין רכב השחקן לבין שאר רכבי הכביש, ישנה פונקציה ייעודית אשר בודקת אם קיימת חפיפה בין שטחי המלבנים של רכב השחקן ובין כל שאר הרכבים. הפונקציה עושה זאת על ידי בדיקה של ארבעה מצבים בהם שתי צלעות בלבד של הרכב נמצאות בתוך מלבן הרכב השני: צלע שמאלית ועליונה, שמאלית ותחתונה, ימנית ועליונה, ימנית ותחתונה. הפונקציה מורידה כ-10 פיקסלים מכל צלע של מלבן המכונית על מנת להתעלם משטחים ריקים של התמונה כמו הפינות והמראות הבולטות.

רשימת הפונקציות

```
def set_background():
    '''Function draws the background to the screen, thereby
erasing the screen.
    No input and no output.'''

def add_background(trees, cows, tree_density, cow_density):
    '''Function adds more elements such as trees to the top of
the screen.
    Input: lists which hold all trees and cows and their
density.
    Output: updated lists which hold all trees and cows.'''

def move_background(trees, cows, speed):
    '''Function moves all background elements such as trees
down at a certain speed while removing elements out of the
screen.
    Input: lists which hold all trees and cows and the number
of pixels to move.
    Output: updated lists which hold all trees and cows.'''

def change_background(trees, cows, speed, tree_density,
cow_density):
    '''Function moves all background elements down and creates
new ones.
    Input: lists which hold all trees and cows, speed to be
moved and the density of which the trees and cows will appear
on the screen.
    Output: updated lists of the trees and cows.'''

def create_object(object_list, object_height):
    '''Function created an individual object at a random spot
in the top of the screen while not interrupting the visibility
of the road.
    Input: list which holds all the objects and the height of
the object.
    Output: updated list which holds all the objects.'''
```

```
def set_lines():
    '''Function creates a new list which holds all positions
of the road lines.
    Input: none.
    Output: a list which holds all positions of the road
lines.'''

def move_lines(lines,speed):
    '''Function moves all road lines downward at a certain
speed.
    Input: a list which holds all positions of the road lines
and the number of pixels to move.
    Output: an updated list which holds all positions of the
road lines.'''

def move_lanes(car, move_to_lane):
    '''Function moves a car horizontally to a certain lane.
    Input: a car list which holds position of the car, model
and size and a lane to be moved to.
    Output: an updated car list which holds position of the
car, model and size.'''

def move_car(car,speed):
    '''Function moves a single car vertically at a certain
speed.
    Input: a car list which holds position of the car, model
and size and the number of pixels to move.
    Output: an updated car list which holds position of the
car, model and size.'''

def lane_pos(lane, car_width):
    '''Function returns the x position of a lane number of
which a car would be in the middle of the lane.
    Input: a lane number and the width of a car.
    Output: x position.'''
```

```python
def create_cars(cars, car_models, player_car_height):
    '''Function creates a random car object on the top of the
screen at a random lane only if its valid.
    Input: a list of all car objects, a list of all car models
and the height of the player car.
    Output: an updated list of all car objects.'''

def car_density(cars):
    '''Function determines the density of which the cars will
be created based on how many cars are currently on the road.
    Input: a list of all car objects.
    Output: the density of which car objects will be
created.'''

def car_valid(car_x, cars, player_car_height):
    '''Functions determines whether a car can be created at a
certain lane based on the height of the player car so there
would be no roadblocks for the player car and no cars on top
of each other.
    Input: lane position of the car to be created, a list of
all car objects and the height of the player car.
    Output: a boolean indicating whether the car can be
created.'''

def move_cars(cars, speed, cars_speed):
    '''Function moves all non player cars to make the illusion
they move at a certain speed.
    Input: a list of all car objects, speed of the player car
and speed of all non player cars.
    Output: an updated list of all non player car objects.'''

def is_crash(player_car ,cars):
    '''Function determines whether the player car crashed with
another car.
    Input: a car object of player and a list of all non player
car objects.
    Output: a boolean indicating whether there is a
collision.'''
```

```
def update_car_models(car_models, current_player_car
,updated_player_car):
    '''Function updates available car models.
    Input: a list containing all currently available car
models,
    the old player car model and the new player car model.
    Output: an updated list containing all currently available
car models.'''


def
create_text(t,center,font="Arial",size=72,color=(255,255,0),
bold=False,italic=False):
    '''Function blits text to the screen based on center
position given.
    Input: a text to blitted, center positions to blit the
text, font of the text, size of the text, color of the text, a
boolean wheter the text is in bold and a boolean whether the
text is in italic.
    Output: none.'''


def round_rect(surface, rect, color, rad=20, border=0,
inside=(0,0,0,0)):
    """
    Draw a rect with rounded corners to surface.  Argument rad
can be specified to adjust curvature of edges (given in
pixels).  An optional border width can also be supplied; if
not provided the rect will be filled. Both the color and
optional interior color (the inside argument) support alpha.
    """


def _render_region(image, rect, color, rad):
    """Helper function for round_rect."""
```

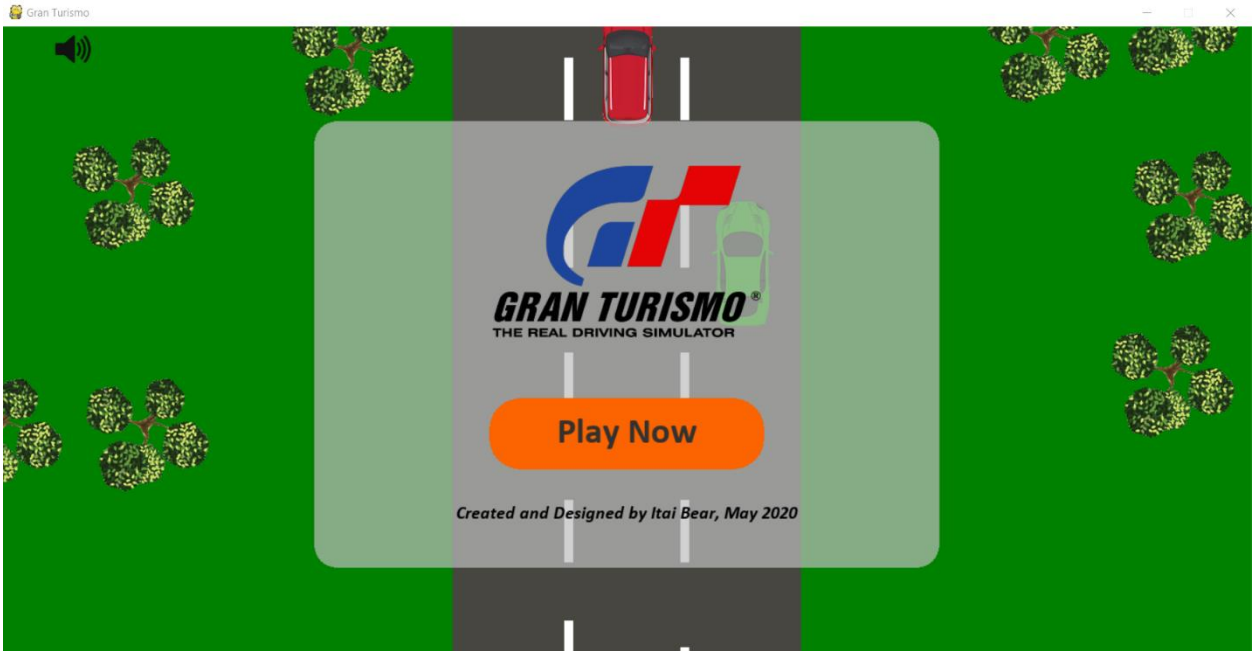
```
def game_menu(menu_center_pos, in_out, speed,
main_or_end,rank=False,score=0):
    '''Functions blits to the screen the game menus.

    Input: the center position of menu, whether the menu is
sliding in or out, speed of the menu slide, whether it is the
main menu or end menu, a boolean whether the player succeded
in getting to the leaderboard and the score.

    Output: the center position of the menu, whether the menu
is sliding in or out, speed of the menu slide and button
rectangles.


    in_out: 0 - no slide, 1 - slide in from the top, 2 - slide
out from below, 3 - slide out from the top, -1 - menu out of
the screen

    main_or_end: 1 - main menu, 2 - end menu'''


def side_score(score):
    '''Function blits to the side of the screen the current
score during the game.

    Input: the current score.

    Output: none.'''


def reset_leaderboard():
    '''Function resets the leaderboard text file also creating
new one if it doesn't exist.

    No input and no output.'''


def find_score_rank(score):
    '''Function finds the rank in the leaderboard of the
score.

    Input: score.

    Output: the rank in the leaderboard of False if it didn't
get in.'''
```

```
def update_leaderboard(rank, name, score):
    '''Recursive function updates leaderboard ranks based on a
new entry to the leaderboard.
    Input: the rank in the leaderboard, the name of the player
and score received.
    Output: none.'''

def update_leaderboard_name(rank,name,score):
    '''Function updates player's name and score based on rank
in the leaderboard.
    Input: rank entry to be updated, a name to update and a
score to update.
    Output: none.'''

def reset_vehicles(finish, trees, cows, lines, cars):
    '''Function is a minor phase in the game which ressets all
cars on the road elegantly by moving the background fast to
make the illusion it moved to a different part of the road.
    Input: a boolean indicating whether the exit button was
pressed, a list of all trees, a list of all cows, a list of
all road lines and a list of all cars.
    Output: a boolean indicating whether the exit button was
pressed.'''

def pause(finish):
    '''Function pauses the game and resumes after a any key
was pressed.
    Input: a boolean indicating whether exit button was
pressed.
    Output: a boolean indicating whether exit button was
pressed.'''

def mute(muted):
    '''Functions mutes and unmutes the music in the game.
    Input: a boolean indicating whether the game was muted
beforehand.
    Output: a boolean indicating whether the game is currently
muted.'''
```

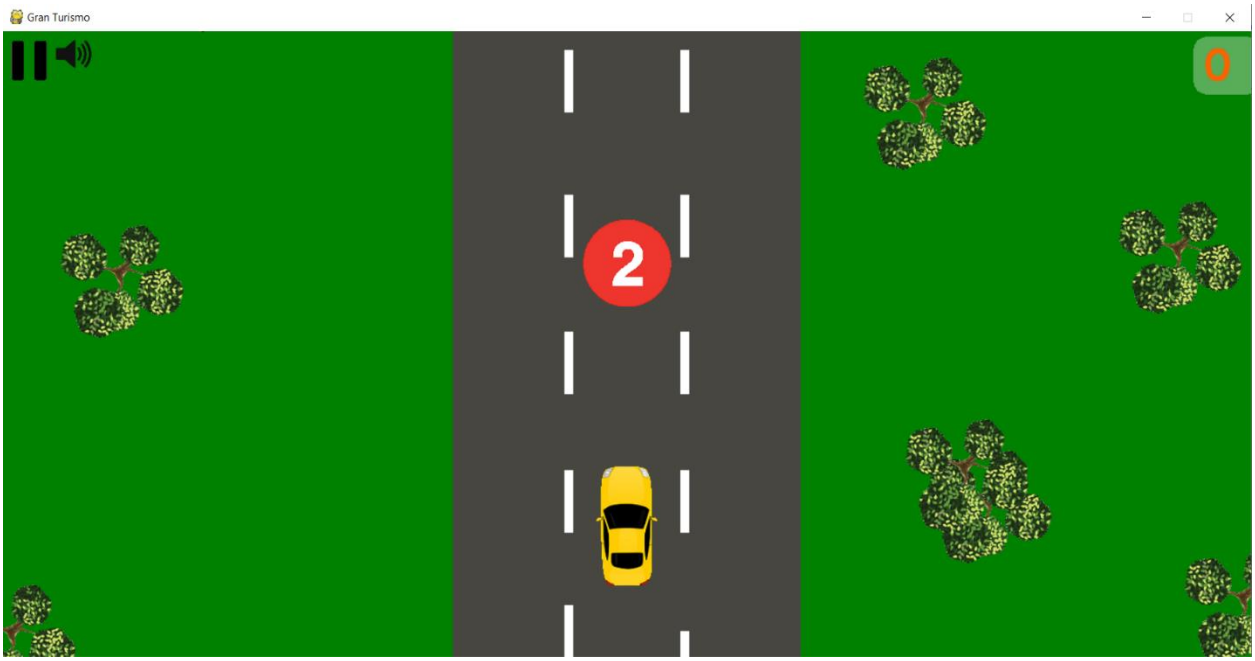
```python
def get_keyboard_language():
    """
    Gets the keyboard language in use by the current
    active window process.
    """


def main():
    '''Function is the main function where the main loop takes
place and all game is managed.
    No input and no output.'''
```

דוגמאות הרצה

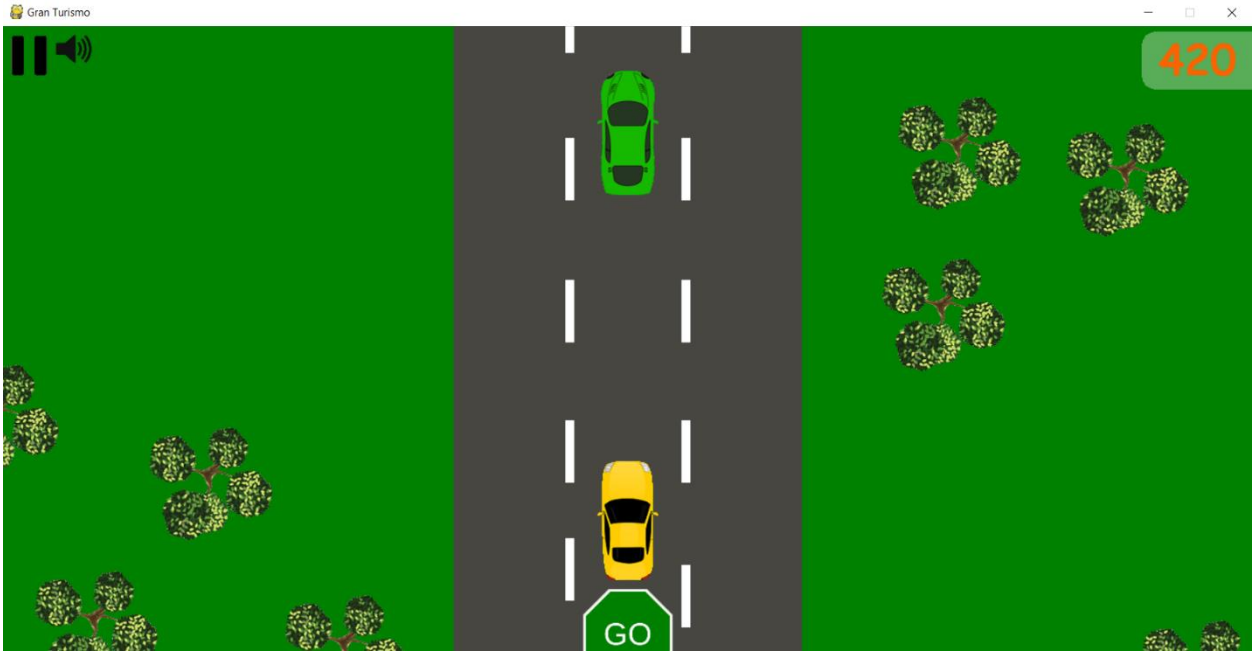
מסך הפתיחה והתפריט הראשי –



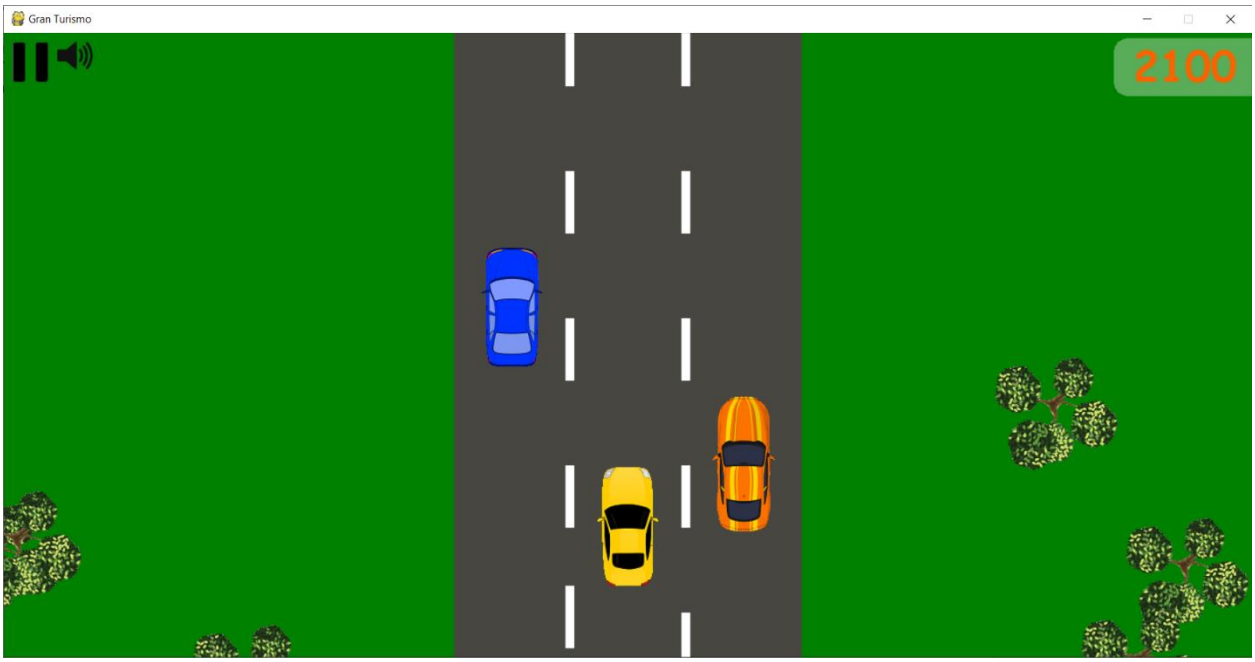
תחילת המשחק והספירה לאחור –

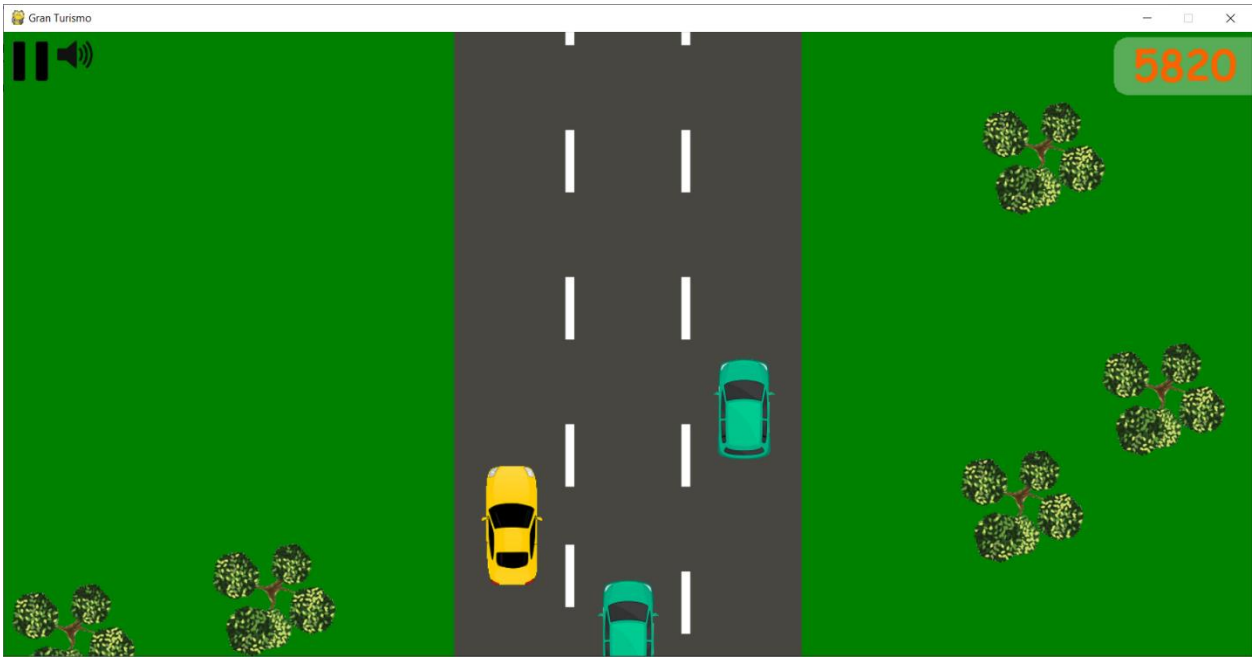


תחילת המשחק וסיום הספירה לאחור -

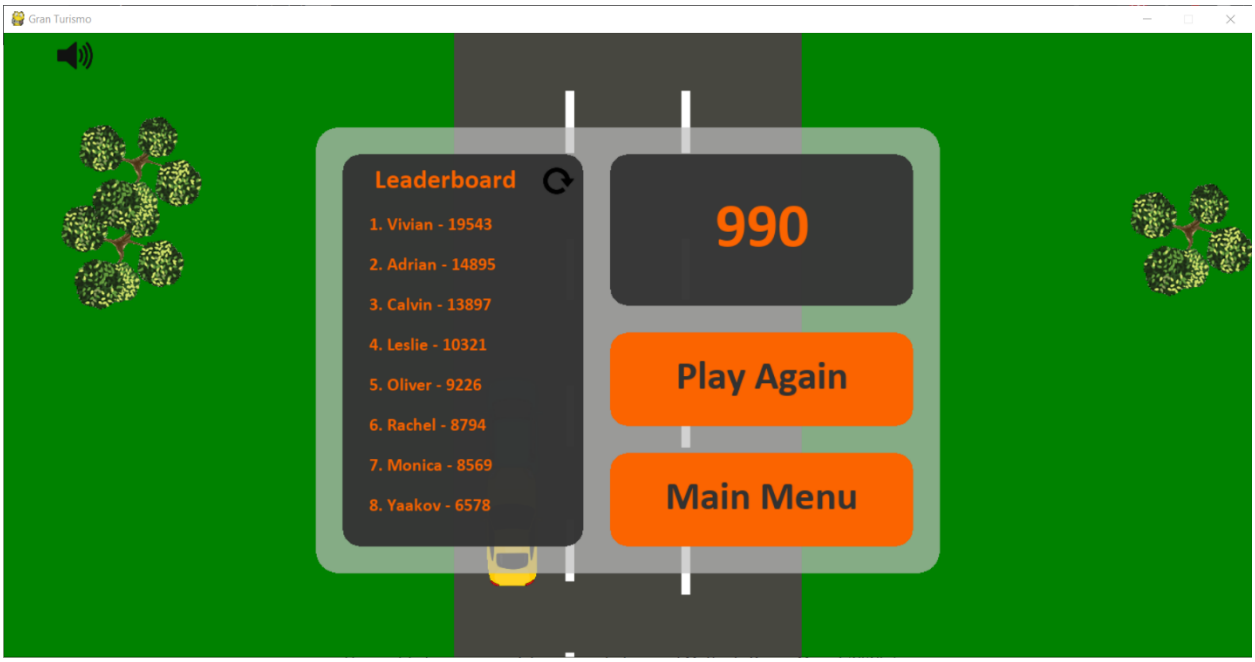


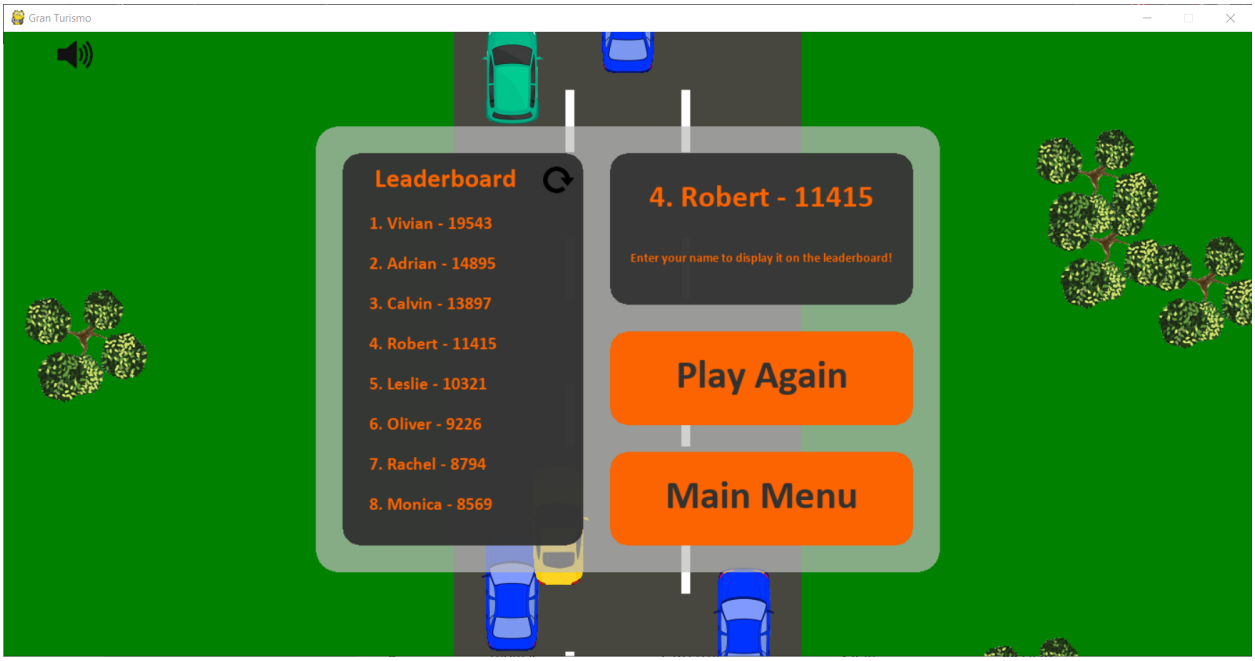
מהלך המשחק –





תפריט סיום המשחק –





מסך השהיית המשחק –