ENGF0002 Design and Professional Skills: Assignment 3 (Frogger)

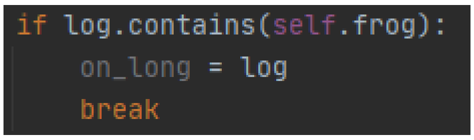
Bug 1: Cannot land on a turtle (player dies)

Description: When trying to move onto one of the turtles in the river, the player dies instead of moving onto it.

What should happen: Should be able to step on the turtle, and move with the turtle as it moves left

How to reproduce: Attempt to move onto a turtle

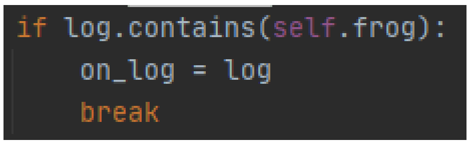
Cause of bug:



```
if log.contains(self.frog):
    on_long = log
    break
```

Typo: ‘on_long’ instead of ‘on_log’

Fixed code:



```
if log.contains(self.frog):
    on_log = log
    break
```

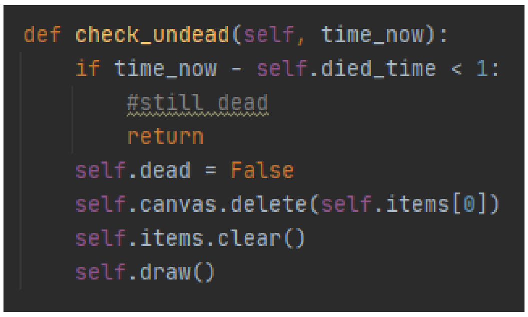
Bug 2: Doesn't reset position after death

Description: Once the player has died, it will stay in the same position where it just died, hence causing it to repeatedly die as it is still in a position that causes death (e.g. hitting a car)

What should happen: Frog should return to its starting location at the bottom of the screen with one less life

How to reproduce: Move frog so it dies (e.g. jump into a car). After the initial death, it will stay in the same spot and repeatedly die as a result.

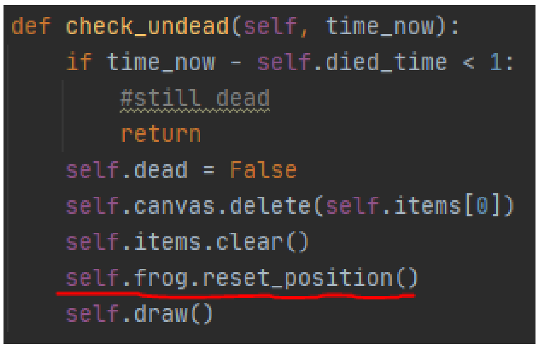
Cause of bug:



```python
def check_undead(self, time_now):
    if time_now - self.died_time < 1:
        #still dead
        return
    self.dead = False
    self.canvas.delete(self.items[0])
    self.items.clear()
    self.draw()
```

Upon dying and respawning, it does not change the position of the frog object, hence it will be redrawn in the same spot that it died (and hence will die again)

Fixed code:



```python
def check_undead(self, time_now):
    if time_now - self.died_time < 1:
        #still dead
        return
    self.dead = False
    self.canvas.delete(self.items[0])
    self.items.clear()
    self.frog.reset_position()
    self.draw()
```

Resets position of the frog object after dying/before being redrawn on the screen

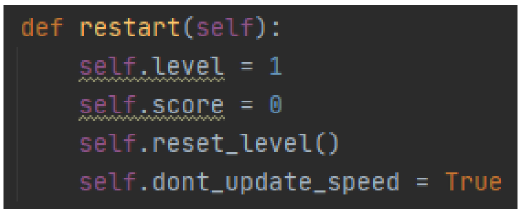
Bug 3: Game does not reset correctly after all lives are lost

Description: Once the player loses all their lives and is prompted to press “r” to reset the game, after the reset the player cannot be moved and none of the cars/logs ect. (moving objects) also do not move

What should happen: The game should be reset to the same state as if it had just been opened, with a fully mobile and functional player frog and moving obstacles on screen.

How to reproduce: Lose all player lives (e.g. running into cars) and press r when prompted to restart the game. Upon pressing r, the player location is reset but the frog cannot be moved/nothing is moving on screen.

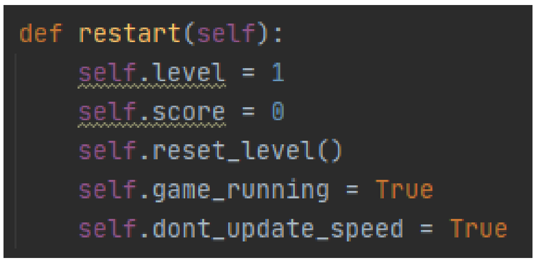
Cause of bug:



```
def restart(self):
    self.level = 1
    self.score = 0
    self.reset_level()
    self.dont_update_speed = True
```

“game_running” variable not set to “true”, hence the game does not properly restart

Fixed code:



```
def restart(self):
    self.level = 1
    self.score = 0
    self.reset_level()
    self.game_running = True
    self.dont_update_speed = True
```

Game_running now set to true after a restart

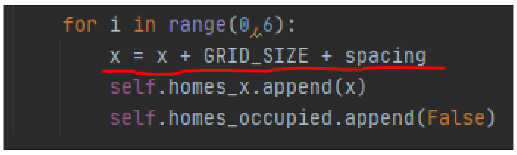
Bug 4: Player dies after reaching leftmost home

Description: When landing on the blue area marking the home furthest left on the screen, the player will die instead of it registering as being home

What should happen: Should give level completed indication (not die)

How to reproduce: Go to leftmost blue home square

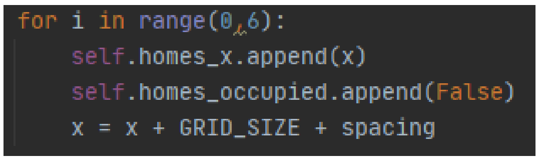
Cause of bug:



```
for i in range(0,6):
    x = x + GRID_SIZE + spacing
    self.homes_x.append(x)
    self.homes_occupied.append(False)
```

Error in “create_homes” function – increases x by the space between homes before appending it. As x (before this function) is set to the position of home 1, the first recorded home position will be home 2

Fixed code:



```
for i in range(0,6):
    self.homes_x.append(x)
    self.homes_occupied.append(False)
    x = x + GRID_SIZE + spacing
```

Changed order so it increases after appending the first house

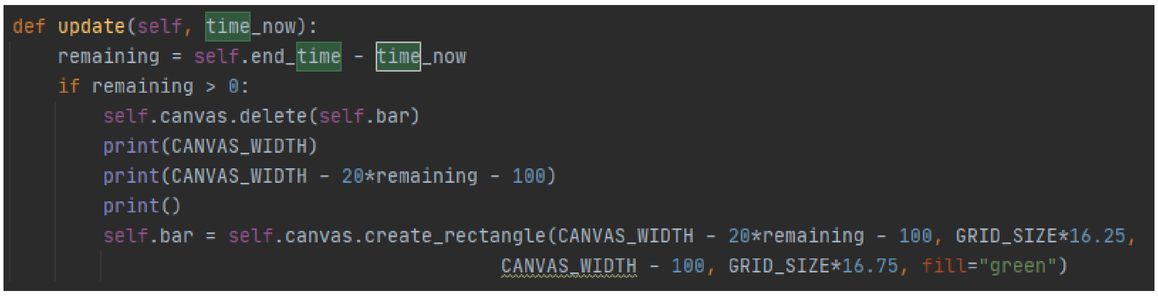
Bug 5: Timer bar goes off left hand side of the screen

Description: Some of timer is not visible to the player as it goes off the left side of the screen

What should happen: Full timer should be visible

How to reproduce: Look at bottom of screen – green timer bar goes off the left side of the screen

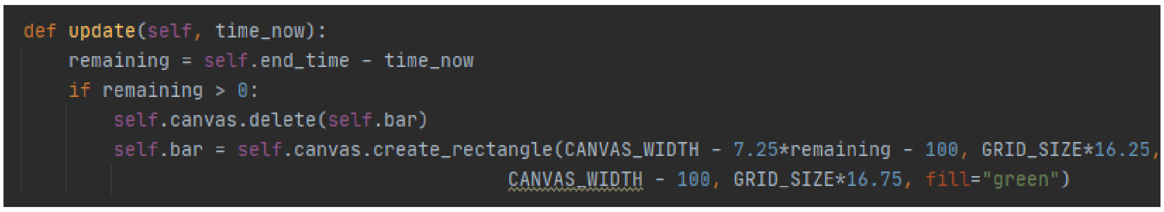
Cause of bug:



```python
def update(self, time_now):
    remaining = self.end_time - time_now
    if remaining > 0:
        self.canvas.delete(self.bar)
        print(CANVAS_WIDTH)
        print(CANVAS_WIDTH - 20*remaining - 100)
        print()
        self.bar = self.canvas.create_rectangle(CANVAS_WIDTH - 20*remaining - 100, GRID_SIZE*16.25,
                                                CANVAS_WIDTH - 100, GRID_SIZE*16.75, fill="green")
```

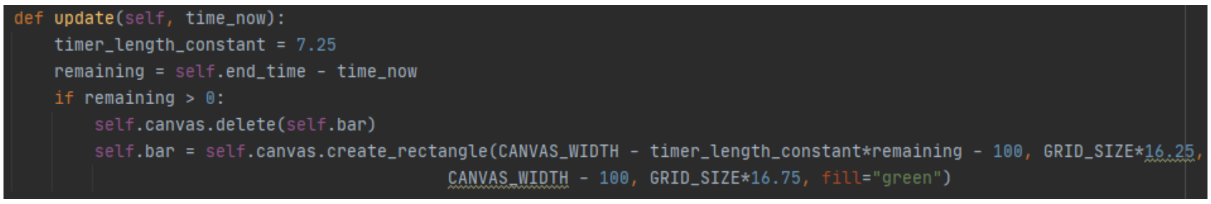
In the “update” function in time_view the length of the bar is given by “CANVAS_WIDTH-100-20*remaining” however 20*remaining is a very large value, hence the length of the bar is a very large negative value

Fixed code:



```python
def update(self, time_now):
    remaining = self.end_time - time_now
    if remaining > 0:
        self.canvas.delete(self.bar)
        self.bar = self.canvas.create_rectangle(CANVAS_WIDTH - 7.25*remaining - 100, GRID_SIZE*16.25,
                                                CANVAS_WIDTH - 100, GRID_SIZE*16.75, fill="green")
```

Changed the 20* to 7.25* so the bar now fits at the bottom of the screen



```python
def update(self, time_now):
    timer_length_constant = 7.25
    remaining = self.end_time - time_now
    if remaining > 0:
        self.canvas.delete(self.bar)
        self.bar = self.canvas.create_rectangle(CANVAS_WIDTH - timer_length_constant*remaining - 100, GRID_SIZE*16.25,
                                                CANVAS_WIDTH - 100, GRID_SIZE*16.75, fill="green")
```

As a further improvement, made the 7.25 constant a variable to avoid magic numbers (rest of self.bar could be improved in similar way)

Bug 6: Can travel off the bottom of the screen

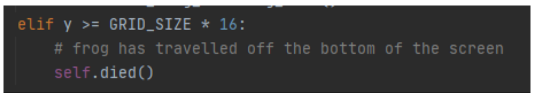
Description: If (instead of travelling up the screen) the player travels downwards, there is no limit on how far the player can go.

What should happen: Player should not be able to travel downwards off the screen (should die if going off screen)

How to reproduce: Move downwards off the screen – player can freely do this, there are no limits

Cause of bug: No code to check for If the frog's y position exceeds a certain level (goes off the bottom of the screen)

Fixed code:



```
elif y >= GRID_SIZE * 16:
    # frog has travelled off the bottom of the screen
    self.died()
```

Added if statement to check for if it equals/exceeds 16 (where 16 = below where the frog initially starts)

Bug 7: Game does not end when timer reaches 0

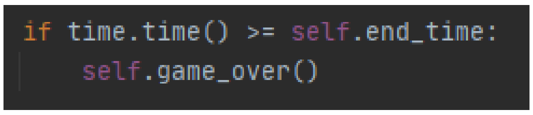
Description: Once the timer runs out, the game doesn't actually end (continues running)

What should happen: Game should end once timer reaches 0

How to reproduce: Wait until timer runs out – game will continue as usual

Cause of bug: No code to check what the time was and trigger game over based on time

Fixed code:



```python
if time.time() >= self.end_time:
    self.game_over()
```

Added "if" statement to "check_frog" function in the frog model, game over now triggers when time is equal to or past the end time.