

## Written Questions - Section 2

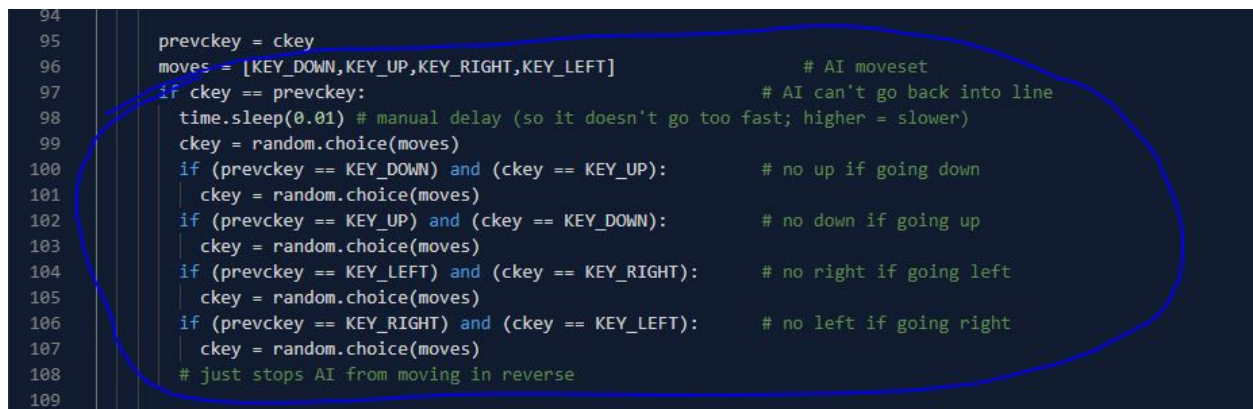
2a.

The coding language used in this project is Python 2.7. The game itself is a Python version of the classic game Tron. This version is the normal 2D top down version, but uses symbols instead of colors. My video demonstrates the game in progress. As the game begins, the player moves up and the AI moves down, although this may change quickly based on the while loop's speed. The player loses by colliding with its own, or the AI's, line; the AI is allowed to go through its own line, for longer gameplay, but otherwise follows the same rules as the player. Going through the borders teleports you, or the AI, to the opposite side. There is a score, in seconds, for how long you have survived.

2b.

Most of the code was functioning to begin with, but my edits and new lines of code took a few trial and error tests to use or function correctly. As most of the code functions within a while loop, there were times where my code did not work and I had to reload the page. Once I got whatever I wanted partially working, I would move onto the next lines of code to improve the previous ones, like getting my AI to turn randomly. While I did get help from my friend to see if my code involving the time package would work, my coding was mostly independent.

2c.



```
94
95     prevkey = ckey
96     moves = [KEY_DOWN, KEY_UP, KEY_RIGHT, KEY_LEFT]
97     if ckey == prevkey:
98         time.sleep(0.01) # manual delay (so it doesn't go too fast; higher = slower)
99         ckey = random.choice(moves)
100         if (prevkey == KEY_DOWN) and (ckey == KEY_UP):
101             ckey = random.choice(moves)
102         if (prevkey == KEY_UP) and (ckey == KEY_DOWN):
103             ckey = random.choice(moves)
104         if (prevkey == KEY_LEFT) and (ckey == KEY_RIGHT):
105             ckey = random.choice(moves)
106         if (prevkey == KEY_RIGHT) and (ckey == KEY_LEFT):
107             ckey = random.choice(moves)
108         # just stops AI from moving in reverse
109
```

This code's purpose is to stop the AI from moving in reverse. While it is somewhat useless with line 98 commented out, it checks if the AI's next movement option is going in reverse to its current direction and does not allow that if so. It functions independently by choosing the AI's next move, but cooperates with the code to make the AI survive longer. By this I mean that the AI crashes if it hits a line, not its own thanks to line 98, and the player can be defeated by the crashing into any line.

2d.

```
94
95     prevkey = ckey
96     moves = [KEY_DOWN,KEY_UP,KEY_RIGHT,KEY_LEFT]           # AI moveset
97     if ckey == prevkey:                                     # AI can't go back into line
98         time.sleep(0.01) # manual delay (so it doesn't go too fast; higher = slower)
99         ckey = random.choice(moves)
100     if (prevkey == KEY_DOWN) and (ckey == KEY_UP):         # no up if going down
101         ckey = random.choice(moves)
102     if (prevkey == KEY_UP) and (ckey == KEY_DOWN):         # no down if going up
103         ckey = random.choice(moves)
104     if (prevkey == KEY_LEFT) and (ckey == KEY_RIGHT):      # no right if going left
105         ckey = random.choice(moves)
106     if (prevkey == KEY_RIGHT) and (ckey == KEY_LEFT):      # no left if going right
107         ckey = random.choice(moves)
108     # just stops AI from moving in reverse
109
110     # increase AI length
```

In these lines, the values assigned to the arrow keys are compared to the values of the new move options. As explained in the curses module page, each key on a keyboard has an integer value so the computer can mathematically compare these values. Besides the internal code behind this code, these lines logically reroll the AI's current movement to avoid moving in reverse into itself.