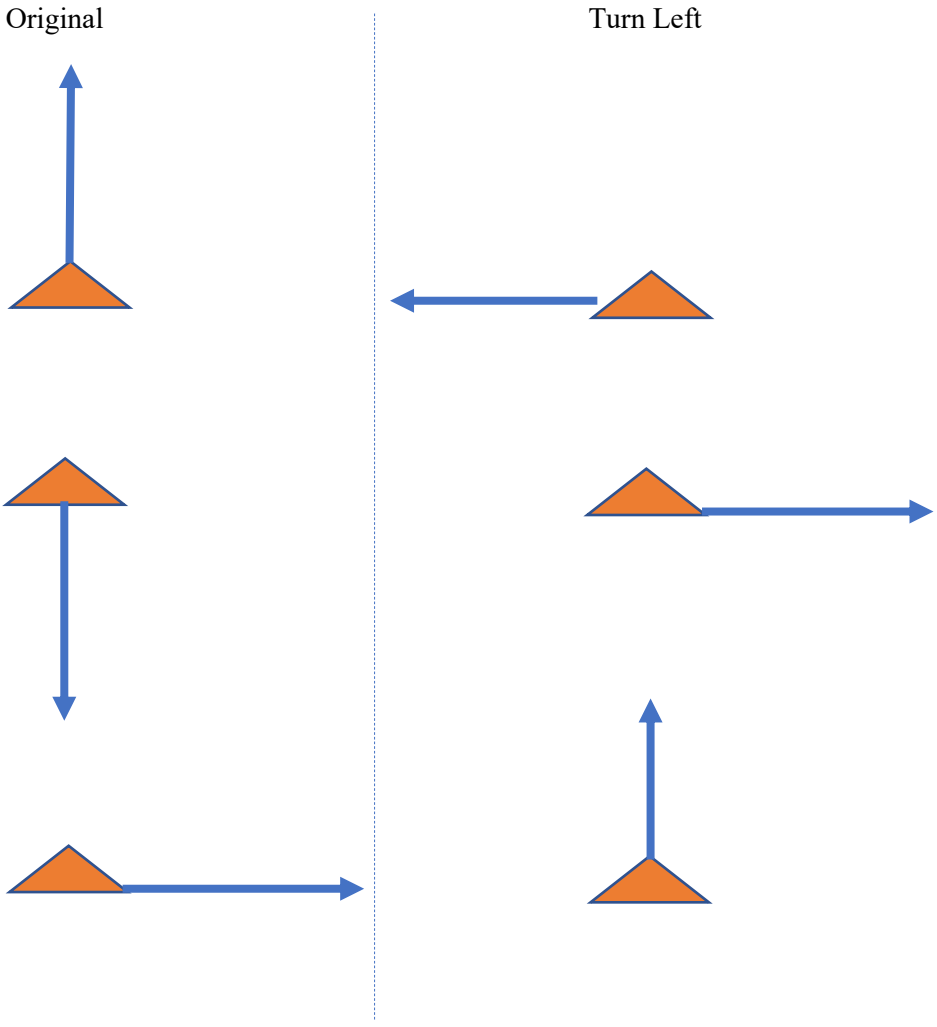
HW3 V2 FAQ

Please see answers to FAQ below. Anything in quotes comes from the HW3 V2 description.

- What happens if our car tries to go off grid?
    - In this case, your car will remain in whatever cell it was in when it tried to go off grid. You will then receive whatever penalty Is associated with that cell.
- In the starting position, do we have a value of 0 or -1?
    - 0, you only receive a value when you make a move
- Do we stop if we hit an obstacle? Or do we continue trying to reach the goal?
    - No, you continue trying to reach the goal
- Using seeds 0-9 works, but 1-10 does not. Which seeds should we use?
    - 0-9
- What do turn_left() and turn_right() mean?
    - turn_left(): Take the move rotated 90 degrees counterclockwise from your chosen move.
      turn_right(): Take the move rotated 90 degrees counterclockwise from your chosen move.
    - Here are some visual examples:



- Do the probabilities of making a wrong move change if we are next to a wall or in a corner?
    - No, if your car tries to go off grid it will behave as described earlier in this FAQ.
- The values of the corrected outputs on DEN are different from the outputs on Vocareum
    - This has been corrected
- What do the values in Hint #3 mean?
    - This is for you to think about and implement. Think about the algorithms we've studied thus far, and ask yourself if this information would be helpful to any of them.
- What indentation level should "k+=1" be at in the sample code for simulation.
    - It should just be inside the while loop. Here is a clearer version:

```
for i in range(len(cars)):
        for j in range(10):
                pos = cars[i]
                np.random.seed(j+1)
                swerve = np.random.random_sample(1000000)
                k=0
                while pos != ends[i]:
                        move = policies[i][pos]
                                if swerve[k] > 0.7:
                                        if swerve[k] > 0.8:
                                                if swerve[k] > 0.9:
                                                        move = turn_left(turn_left(move))
                                                else:
                                                        move = turn_left(move)
                                        else:
                                                move = turn_right(move)
                        k+=1
```

when prob ≤ 0.7:
original direction

when 0.7 < prob ≤ 0.8:
counterclockwise 90°

when 0.8 < prob ≤ 0.9:
clockwise 90°

when 0.9 < prob ≤ 1:
180°