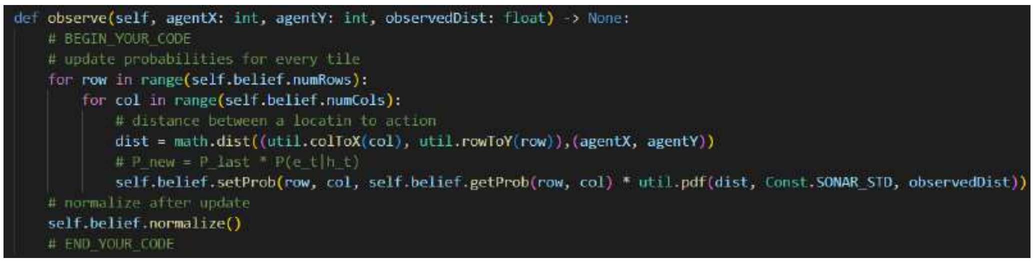
Homework 5: Car Tracking

Part I. Implementation (15%):

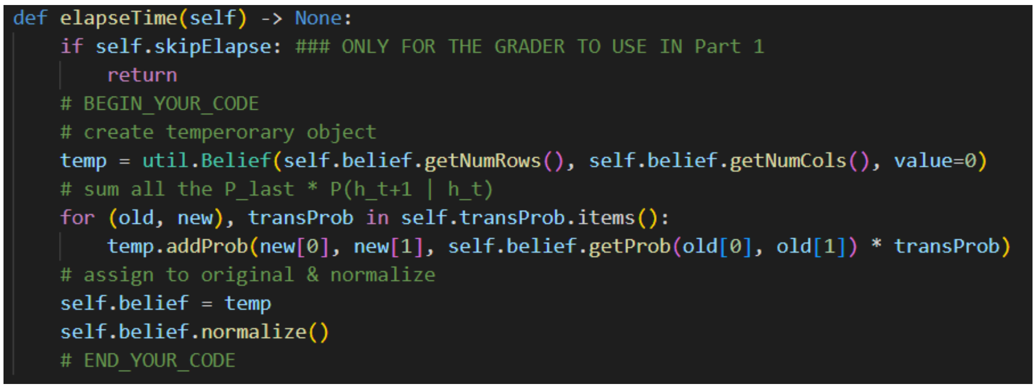
(explanation in comment)

Part. 1



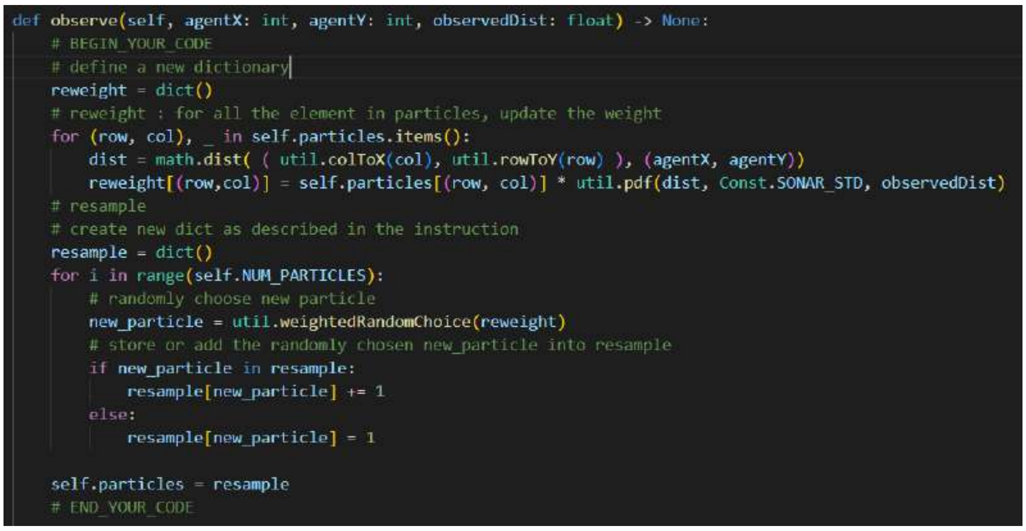
```python
def observe(self, agentX: int, agentY: int, observedDist: float) -> None:
    # BEGIN_YOUR_CODE
    # update probabilities for every tile
    for row in range(self.belief.numRows):
        for col in range(self.belief.numCols):
            # distance between a locatin to action
            dist = math.dist((util.colToX(col), util.rowToY(row)),(agentX, agentY))
            # P_new = P_last * P(e_t|h_t)
            self.belief.setProb(row, col, self.belief.getProb(row, col) * util.pdf(dist, Const.SONAR_STD, observedDist))
    # normalize after update
    self.belief.normalize()
    # END_YOUR_CODE
```

Part. 2



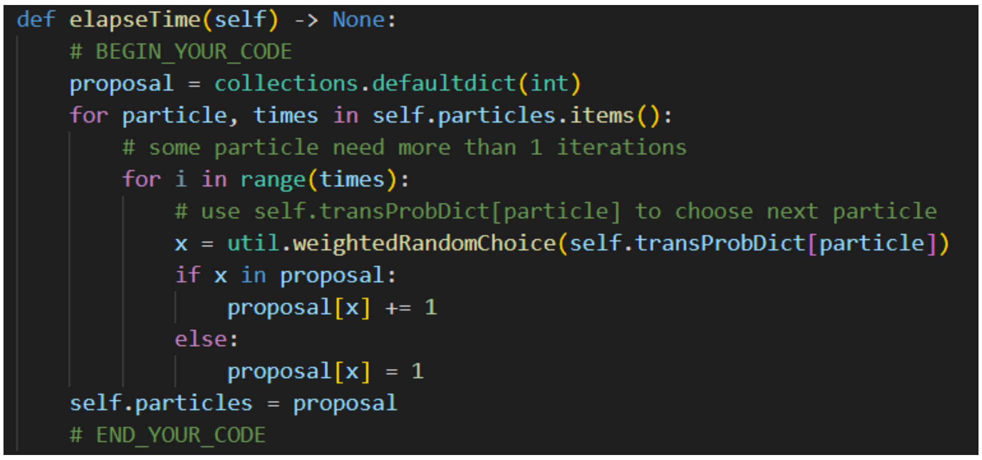
```python
def elapseTime(self) -> None:
    if self.skipElapse: ### ONLY FOR THE GRADER TO USE IN Part 1
        return
    # BEGIN_YOUR_CODE
    # create temperorary object
    temp = util.Belief(self.belief.getNumRows(), self.belief.getNumCols(), value=0)
    # sum all the P_last * P(h_t+1 | h_t)
    for (old, new), transProb in self.transProb.items():
        temp.addProb(new[0], new[1], self.belief.getProb(old[0], old[1]) * transProb)
    # assign to original & normalize
    self.belief = temp
    self.belief.normalize()
    # END_YOUR_CODE
```

Part. 3-1



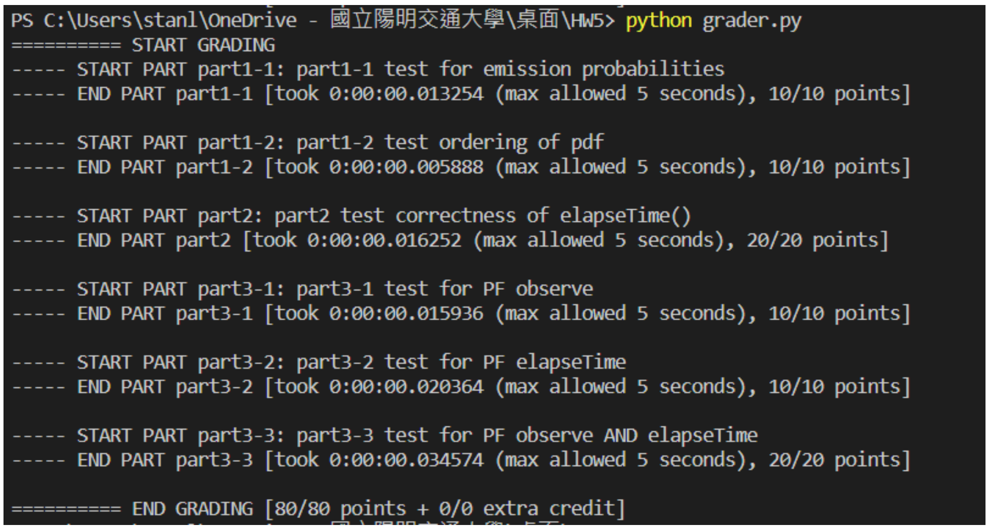
```python
def observe(self, agentX: int, agentY: int, observedDist: float) -> None:
    # BEGIN_YOUR_CODE
    # define a new dictionary
    reweight = dict()
    # reweight : for all the element in particles, update the weight
    for (row, col), _ in self.particles.items():
        dist = math.dist( ( util.colToX(col), util.rowToY(row) ), (agentX, agentY))
        reweight[(row,col)] = self.particles[(row, col)] * util.pdf(dist, Const.SONAR_STD, observedDist)
    # resample
    # create new dict as described in the instruction
    resample = dict()
    for i in range(self.NUM_PARTICLES):
        # randomly choose new particle
        new_particle = util.weightedRandomChoice(reweight)
        # store or add the randomly chosen new_particle into resample
        if new_particle in resample:
            resample[new_particle] += 1
        else:
            resample[new_particle] = 1

    self.particles = resample
    # END_YOUR_CODE
```

Part. 3-2



```python
def elapseTime(self) -> None:
    # BEGIN_YOUR_CODE
    proposal = collections.defaultdict(int)
    for particle, times in self.particles.items():
        # some particle need more than 1 iterations
        for i in range(times):
            # use self.transProbDict[particle] to choose next particle
            x = util.weightedRandomChoice(self.transProbDict[particle])
            if x in proposal:
                proposal[x] += 1
            else:
                proposal[x] = 1
    self.particles = proposal
    # END_YOUR_CODE
```

Result



```
PS C:\Users\stanl\OneDrive - 國立陽明交通大學\桌面\HW5> python grader.py
========== START GRADING
----- START PART part1-1: part1-1 test for emission probabilities
----- END PART part1-1 [took 0:00:00.013254 (max allowed 5 seconds), 10/10 points]

----- START PART part1-2: part1-2 test ordering of pdf
----- END PART part1-2 [took 0:00:00.005888 (max allowed 5 seconds), 10/10 points]

----- START PART part2: part2 test correctness of elapseTime()
----- END PART part2 [took 0:00:00.016252 (max allowed 5 seconds), 20/20 points]

----- START PART part3-1: part3-1 test for PF observe
----- END PART part3-1 [took 0:00:00.015936 (max allowed 5 seconds), 10/10 points]

----- START PART part3-2: part3-2 test for PF elapseTime
----- END PART part3-2 [took 0:00:00.020364 (max allowed 5 seconds), 10/10 points]

----- START PART part3-3: part3-3 test for PF observe AND elapseTime
----- END PART part3-3 [took 0:00:00.034574 (max allowed 5 seconds), 20/20 points]

========== END GRADING [80/80 points + 0/0 extra credit]
```

Part II. Question answering (5%):

The most time I spent is on understanding the concept and the functions, but once I understood what I should do, the rest is not so difficult.

The only thing I spent on coding is at part3-2. I used dict() at first and it gave rise to the value error saying the index is not found. I solve it by using collections.defaultdict(int), which will have default value 0 for undetermined value.