**Virtual Pi2Go Programming: WS22 Sample Answers**

**Question 1:** No, just knowing the best action overall isn’t enough. We want the robot to different things depending upon whether it is accurately placed at the edge of the oval or is in the middle of the black and so on.

**Exercise 1:**

def action\_reward():

 action\_rewards = {}

 for i in range(0, 2):

 for j in range(0, 2):

 for k in ('left', 'right', 'reverse', 'forward'):

 action\_rewards[((i, j), k)] = 0

 return action\_rewards

**Exercise 2:**

def action\_reward(action\_list, default):

 action\_rewards = {}

 for i in range(0, 2):

 for j in range(0, 2):

 for k in (action\_list):

 action\_rewards[((i, j), k)] = default

 return action\_rewards

**Exercise 3:** Note that this program has been tidied up a bit using functions.

import simclient.simrobot as pi2go

import time, random

pi2go.init()

actions = ['forward','backward','left','right']

def action\_reward(action\_list, default):

 action\_rewards = {}

 for i in range(0, 2):

 for j in range(0, 2):

 for k in (action\_list):

 action\_rewards[((i, j), k)] = default

 return action\_rewards

def execute\_action(action):

 if (action == "forward"):

 pi2go.forward(10)

 elif (action == "backward"):

 pi2go.reverse(10)

 elif (action == "left"):

 pi2go.spinLeft(10)

 elif (action == "right"):

 pi2go.spinRight(10)

 else:

 pi2go.stop()

 time.sleep(3)

reward\_dictionary = action\_reward(actions, 0)

total\_attempts = action\_reward(actions, 0)

rewards = {(1, 1):1, (1, 0):2, (0, 1):0, (0, 0):1}

count = 0

while (count < 50):

 action = random.choice(actions)

 state = (pi2go.irLeftLine(), pi2go.irRightLine())

 execute\_action(action)

 reward\_dictionary[(state, action)] = reward\_dictionary[(state, action)] + rewards[(pi2go.irLeftLine(), pi2go.irRightLine())]

 total\_attempts[(state, action)] = total\_attempts[(state, action)] + 1

 count = count + 1

pi2go.stop()

print(reward\_dictionary)

for key in reward\_dictionary:

 total = total\_attempts[key]

 if (total == 0):

 print("Never attempted " + str(key))

 else:

 print("Average reward for " + str(key) + " is " + str(reward\_dictionary[key]/total\_attempts[key]))

**Question 2:** Probably yes, though they could get lucky.

**Question 3:** It should do. If the robot starts a long way from the oval then it is unlikely it will ever be in any of the states where a line sensor is activated.



 University of Liverpool, 2020

This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).