**Virtual Initio Programming: WS21 Sample Answers**

**WS21**

**Question 1:** The keys are forward, backward, left, right and stop

**Question 2:** The values are 3, 2, 5, 3, 2

**Question 3:** The program picks an action at random and executes for a specific duration name 3 seconds for forward, 2 seconds for backward, 5 seconds for left, 3 seconds for right and 2 seconds for stop.

**Exercise 1:**

import simclient.simrobot as initio

import time, random

initio.init()

action\_dictionary = {"forward":3, "backward":2, "left":5, "right":3, "stop":2}

for action in action\_dictionary:

if (action == "forward"):

initio.forward(10)

elif (action == "backward"):

initio.reverse(10)

elif (action == "left"):

initio.spinLeft(10)

elif (action == "right"):

initio.spinRight(10)

else:

initio.stop()

time.sleep(action\_dictionary[action])

**Question 4:** The program selects an action at random. If the action results in either of the line sensors detecting black then it adds one to the value stored for the action in the dictionary. When any action gets a score of more than 3 then it stops.

**Question 5:** If the program starts on the square it terminates really quickly whereas if it starts off the square it runs for a long time until the robot by chance moves onto the square.

**Exercise 2:**

import simclient.simrobot as initio

import time, random

initio.init()

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

action\_dictionary = {"forward":0, "backward":0, "left":0, "right":0, "stop":0}

count = 0

while (count < 20):

action = random.choice(actions)

if (action == "forward"):

initio.forward(10)

elif (action == "backward"):

initio.reverse(10)

elif (action == "left"):

initio.spinLeft(10)

elif (action == "right"):

initio.spinRight(10)

else:

initio.stop()

time.sleep(3)

if (initio.irLeftLine()):

action\_dictionary[action] = action\_dictionary[action] + 1

if (initio.irRightLine()):

action\_dictionary[action] = action\_dictionary[action] + 1

count = count + 1

initio.stop()

score = 0

winner = "no winner"

for key in action\_dictionary:

if (action\_dictionary[key] > score):

score = action\_dictionary[key]

winner = key

print("And the winner is: " + winner)

**Question 6:** The rewards dictionary provides a score which depends upon the values of the two line sensors: 2 if both sensors detect black, 1 if only one sensor detects black and 0 if neither sensor detects black. These scores are then added to the action scores each time an action is selected at random.

**Exercise 3:**

import simclient.simrobot as initio

import time, random

initio.init()

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

dictionary = {"forward":0, "backward":0, "left":0, "right":0, "stop":0}

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

while True:

action = random.choice(actions)

if (action == "forward"):

initio.forward(10)

elif (action == "backward"):

initio.reverse(10)

elif (action == "left"):

initio.spinLeft(10)

elif (action == "right"):

initio.spinRight(10)

else:

initio.stop()

time.sleep(3)

dictionary[action] = dictionary[action] + rewards[(initio.irLeftLine(), initio.irRightLine())]

if (dictionary[action] > 3):

break

initio.stop()

print(dictionary)

**Ex21**

**Exercise 1:**

import simclient.simrobot as initio

import time, random

initio.init()

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

rewards = {1:1, 0:0}

action = random.choice(actions)

if (action == "forward"):

initio.forward(10)

elif (action == "backward"):

initio.reverse(10)

elif (action == "left"):

initio.spinLeft(10)

elif (action == "right"):

initio.spinRight(10)

else:

initio.stop()

time.sleep(3)

initio.stop()

reward = rewards[initio.irLeftLine()]

print("The reward is: " + str(reward))

**Exercise 2:**

import simclient.simrobot as initio

import time, random

initio.init()

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

scores = {'forward':0,'backward':0,'left':0,'right':0,'stop':0}

rewards = {1:1, 0:0}

count = 0

while count < 20:

action = random.choice(actions)

if (action == "forward"):

initio.forward(10)

elif (action == "backward"):

initio.reverse(10)

elif (action == "left"):

initio.spinLeft(10)

elif (action == "right"):

initio.spinRight(10)

else:

initio.stop()

time.sleep(3)

reward = rewards[initio.irLeftLine()]

scores[action] = scores[action] + reward

count = count + 1

initio.stop()

print("The rewards are: " + str(scores))

**Exercise 3:**

import simclient.simrobot as initio

import time, random

initio.init()

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

scores = {'forward':0,'backward':0,'left':0,'right':0,'stop':0}

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

count = 0

while count < 20:

action = random.choice(actions)

if (action == "forward"):

initio.forward(10)

elif (action == "backward"):

initio.reverse(10)

elif (action == "left"):

initio.spinLeft(10)

elif (action == "right"):

initio.spinRight(10)

else:

initio.stop()

time.sleep(3)

reward = rewards[(initio.irLeftLine(), initio.irRightLine())]

scores[action] = scores[action] + reward

count = count + 1

initio.stop()

print("The rewards are: " + str(scores))

**Exercise 4:**

import simclient.simrobot as initio

import time, random

initio.init()

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

scores = {'forward':0,'backward':0,'left':0,'right':0,'stop':0}

attempts = {'forward':0,'backward':0,'left':0,'right':0,'stop':0}

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

count = 0

while count < 20:

action = random.choice(actions)

if (action == "forward"):

initio.forward(10)

elif (action == "backward"):

initio.reverse(10)

elif (action == "left"):

initio.spinLeft(10)

elif (action == "right"):

initio.spinRight(10)

else:

initio.stop()

time.sleep(3)

reward = rewards[(initio.irLeftLine(), initio.irRightLine())]

scores[action] = scores[action] + reward

attempts[action] = attempts[action] + 1

count = count + 1

initio.stop()

for action in actions:

if (attempts[action] != 0):

average\_reward = scores[action]/attempts[action]

else:

average\_reward = 0

print("The reward for " + action + " is " + str(average\_reward))



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/).