This is a worksheet for the third COMP3071 lab.

In this session, you'll work on two tasks.

The objective of the first task is to update the existing code to enable a systematic comparison of various parameter settings against the average amount of dirt collected by the robot vacuum cleaner system. In the second task, the goal is to integrate a planning algorithm that can efficiently direct the movement of the robot towards the least costly path between two given points.

Task 2 is,how we can do plotting to see the average

# Task 1

## Task 1

The primary objective of the first task is to improve simpleBot2\_withCounting.py to enable a methodical comparison of various parameter settings concerning the quantity of dirt collected by the robot vacuum cleaner system. Specifically, we want to examine how the number of bots affects the total amount of dirt collected. For each configuration, the experiment needs to be run multiple times.

To achieve this goal, the following modifications can be implemented:

1. Rename the main method as runOneExperiment

* Line 288 and Line 296

1. Update the runOneExperiment method to take a parameter called numberOfBots.

Changes 1

def main(numberOfBots):  
 window = tk.Tk()  
 canvas = initialise(window)  
 registryActives, registryPassives, count = register(canvas, numberOfBots)

1. Modify the register function and its call in the runOneExperiment method to enable the supply of different numberOfBots values.

Changes 1

def register(canvas,noOfBots):  
 registryActives = []  
 registryPassives = []  
 # noOfBots = 1 # Comment out this line

Changes 2

Make sure to comment out the noOfBots on Line 253

1. Instead of just printing the average amount of dirt collected, update the code to extract this information by modifying the moveIt() method.
2. Let's consider how we can achieve this requirement. Previously, the system was set to exit once the following if-statement was fulfilled.

if moves>numberOfMoves:  
 sys.exit()

1. To accomplish this, comment out sys.exit() and replace it with window.destroy() to close the window instead of exiting.
2. After this modification, an element related to the window variable is missing. This element needs to be added to the moveIt() method.

Change 1

def moveIt (canvas,registryActives,

registryPassives,count,moves,window):  
 moves += 1  
 for rr in registryActives:  
 chargerIntensityL, chargerIntensityR = rr.senseCharger(registryPassives)  
 rr.transferFunction(chargerIntensityL, chargerIntensityR)  
 rr.move(canvas,registryPassives,1.0)  
 registryPassives = rr.collectDirt(canvas,registryPassives, count)  
 numberOfMoves = 500  
 if moves>numberOfMoves:  
 print("total dirt collected in",numberOfMoves,"moves is",count.dirtCollected)  
 # sys.exit()  
 window.destroy()  
 canvas.after(50,moveIt,canvas,registryActives,registryPassives,count,moves,window)

1. The runOneExperiment() method also needs to be updated to return the total amount of dirt collected.

Change 1

def runOneExperiment(noOfBots):  
 window = tk.Tk()  
 canvas = initialise(window)  
 registryActives, registryPassives, count = register(canvas,noOfBots)  
 moves = 0  
 moveIt(canvas,registryActives,registryPassives, count, moves,window)  
 window.mainloop()

1. In the updated runOneExperiment() method, add the following argument at the end of the method to return the total amount of dirt collected

return count.dirtCollected

1. Test the code by running the runOneExperiment(numberOfBots) function at the end of the file. Assign any value to numberOfBots. This code should work if all the steps from 1 to 5 are followed.

A. Alternatively, the line can be written as print(runOneExperiment(5)) to print the total amount of dirt collected by 5 bots to the console.

# Task 2

I want the student to create this code, and how they can plot, and save the data

from simpleBot2\_withCounting\_soln import \*  
import pandas as pd  
import matplotlib.pyplot as plt  
from scipy.stats import ttest\_ind  
  
def runSetOfExperiments(numberOfRuns,numberOfBots):  
 dirtCollectedList = []  
 for \_ in range(numberOfRuns):  
 dirtCollectedList.append(runOneExperiment(numberOfBots))  
 return dirtCollectedList  
   
def runExperimentsWithDifferentParameters():  
 *'''  
 The number of robot must be larger than 2  
 :return:  
 '''* resultsTable = {}  
 for numberOfBots in range(1,3):  
 dirtCollected = runSetOfExperiments(1,numberOfBots)  
 resultsTable["robots: "+str(numberOfBots)] = dirtCollected  
 results = pd.DataFrame(resultsTable)  
 print(results)  
 results.to\_excel("data.xlsx")  
 print(ttest\_ind(results["robots: 1"],results["robots: 2"]))  
 print(results.mean(axis=0))  
 results.boxplot(grid=False)  
 plt.show()