Coursework 1 Report

YIT19488399

Delivery date: 15 February 2022

Value: 25% of total

For designing the program of vacuum cleaner, I used and followed instructions from “Coursework 1 Specifications”. I used Python 3 (programming language) and IDE was Jupyter Notebook. I used my computer and I worked in class during lab time, at the University of Roehampton’s library, and in my student room. I asked classmates about any idea how to do the coursework and answers were very vague. Following the recommendations of the module about reading, I checked both books online and find both books very useful, especially: “Artificial Intelligence: a Modern Approach, EBook, Global Edition: A Modern Approach” by Stuart Rusell and Peter Norvig; pages 36, 70 and 143. The second book was: “Introduction to Artificial Intelligence” by Wolfgang Ertel and Nathanael T. Black; was less useful for this Coursework 1, but helped me to understand the logic of my vacuum cleaner design.

My coursework has 3 vacuum cleaner solutions. Starting from a simple and easy one to the final one and more elaborated and professional. The final solution can be improved by optimizing the movement of the cleaner in the matrix (this would be 4th solution). Solution number 1 is very simple. I used reference: <https://github.com/A-safarji/vacuum-cleaner-agent/blob/main/vacuum-cleaner.py>. Images follow:

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Solution number 2 is more elaborated but is not finished because I started coding it when solution Lab2 get published on Moodle. Difficult was to be patient about seeing the solution from Lab 2 from Dr. Y. Gu. And the video from Dr. Araujo where the explanation of the code is given in more detail. After finalising the class on Tuesday 25 of February 2022 at 6:00 pm (UK time), I was unable to access this document/resource neither video. Nothing similar online is available. In this solution, I use graphics and the logic of the algorithm similar to the Lab2 solution. Resources: <https://matplotlib.org/>. While waiting for solution Lab 2, reference <https://notebook.community/pblanc5/Artificial-Intelligence-/Homework2/Homework%202> was helpful because of similarity with Dr. Y. Gu. Images:

Graphical user interface

Description automatically generated

Solution number 3 follows logic from solution Lab2 and previous reference from the Internet (<https://notebook.community/pblanc5/Artificial-Intelligence-/Homework2/Homework%202>). It is a simple algorithm and easy to understand. But because of the use of many functions (13 in total) plus driving block (last cell of code) and importing libraries with initialisation of variables (2 Jupyter notebook code cells in one); it is very easy to get lost in the code. The total number of cells is 15. For a computer is easy because it retains in memory all functions with corresponding variables. While programming, it is much easier to divide all the code into small pieces and this is what I did. While loops are designed in a way that never my program will do an infinite loop. Many comments are included in the code to help to understand my program because it jumps from one block to another frequently. I believe that will be around 1800 steps executed in my code for a simple matrix of size 3x3. Time and space complexity I think increase exponentially when bigger size of the matrix like 5x5. Lab2 solution has about 370 steps for a line of 9 cells in the horizontal position. Reference: <https://pythontutor.com/visualize.html#mode=edit>.

Solution number 3 ask the user to input the size of the matrix. A bug that I leave in code is the possibility to enter any size because is very exciting to see how the agent is cleaning. After, I ask the user for input dirt coordinates. Same procedure to input location of the agent at the beginning of the program. Before starting my agent to clean, I display nicely on the screen to the user the map of the initial situation. Resource used for map: <https://seaborn.pydata.org/generated/seaborn.heatmap.html>. All previous steps are verified by function “input\_row\_col()” and regex: <https://regexone.com/references/python> . The map will display dark blue colour for specks of dirt (3), less dark blue colour when the agent is on dirt (2), and the colour of the agent is light blue (1). White/Transparent is when the cell of the matrix is clean (0). At the end of the cleaning of the agent, the matrix only has one light blue colour (1), because the agent is in the matrix. Agent stop to move and program finish. The reason why my map is composed of rectangular cells is because of the use of the seaborn library (<https://seaborn.pydata.org/generated/seaborn.heatmap.html>). My map now looks much better than the previous solution (Solution 2) using the matplotlib library. Image:

Chart

Description automatically generated

When all input from the user is in my program, the agent starts to move. Function “action(matrix)” start to find the agent coordinates in the matrix using function “action\_location\_cleaner(matrix)”. Firstly it tries to find number 2 because this represents dirt and robot. After, it tries to find number 1 because it represents only robot. “action\_location\_cleaner(matrix)” can not have errors because it gets executed without human intervention, but in case I print to the user “Error! \nSomething wrong! \nRestart the program". When “location\_cleaner” is found, “action(matrix)” invokes the function “action\_new\_action(matrix, location\_cleaner)”. This function represents the movement of the agent in the matrix. The agent can start to move left or right depending on the initial location of the agent. Picture by hand helps to understand how my agent moves. And this hand made scheme help me to write code (if statements with correct indentation). From “action\_new\_action(matrix, location\_cleaner)” function, 3 means move right and 2 means move left.

A piece of paper with writing on it

Description automatically generated with medium confidence

Function responsible for cleaning is “do\_action(matrix, action)”. This function gives instructions about what to do the agent and where to do it. Moreover, this function changes the colour of the blues in the matrix after cleaning. And it uses the “check\_matrix\_to\_dirt(matrix)” function to find dirt on the entire matrix, returning True or False (clean or not clean the matrix) that is used in the condition at the beginning of “do\_action(…)”. In the driving block (last one in the Jupiter notebook), “do\_action(matrix, action)” is in the while loop until ALL the matrix is clean, then the while loop from the driving block finish and the program ends.

Code that is commented “action\_dirt\_search(matrix, location\_cleaner)” is unfinished because my next step is to make the agent looks intelligent. My program moves agent cell by cell in the horizontal direction (left or right), but my program can be improved with the agent moving up and down before finishing the row because dirt is up or down in the neighbour row while moving horizontally the agent. This feature will be great to add, but I need more time to develop it. Pictures and screenshots of Solution3 follow:

Graphical user interface, text

Description automatically generated

Graphical user interface, application

Description automatically generated

A picture containing chart

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, application

Description automatically generated

**Regarding the questions from the task:**

*“o How did you decompose the problem and why?”*

Small functions doing a specific task, except “do\_action(matrix, action)” that do two tasks. Driving code is no function that uses all the previous designed functions.

*“o Have you met any difficulties/bugs during this coursework, and how did you tackle it?”*

A lot of difficulties because is long code and code jumps from one part to another. Google and books helped me a lot to design pieces of my program. There is no similar program on the Internet, but everything used is available on the Internet.

*“o What did you think is the most difficult part of this coursework?”*

Have the initial idea structure of the program. The code is simple, but implementing the logic is a little bit difficult, especially for disorganised people.

*“o What have you learnt from investigating this problem?”*

The “Divide and conquer” approach use many small functions. If loops are very powerful if right use and adequate indentation. Whiles loops same as ifs.

*“o Which part(s) in your code design is your favourite and why?”*

Designing the matrix because is nice graphically. And this function is only 3 lines of code. Solution1 and Solution2 were very easy to implement but is not vacuum cleaner code.

*“o How did you improve the performance of your cleaner? If you had no time limit, how would you further improve your program?”*

My agent is not optimized because it moves without checking dirt around it. And another obvious problem is that it jumps from the end of the matrix to the [0,0] cell in the matrix. This can not happen in real life.

*“o Any references you might have used to complete the task.”*

All references are included previously. Internet the most useful resource.

####################################### END ########################################