

**e-Yantra Robotics Competition - 2020-21**

**Nirikshak Bot**

**Task 4C - Theme and Implementation Analysis**

**NB\_887**

|  |  |
| --- | --- |
| **Team Leader Name** | Kallol Saha |
| **College** | Manipal Institute of Technology, Manipal |
| **Team Leader Email** | kallolsaha1752001@gmail.com |
| **Date** | 14th January, 2021 |

**Scope**

**Q1. State the scope of the theme assigned to you. (5)**

< Teams should briefly explain in their own words the theme assigned. What in your opinion is the purpose of such an application? You may use figures/diagrams to support your answer (Make neat and labelled diagrams).

Answer format: Text - limit: 100 words. >

The Nirikshak Bot can be used in the production line of a company that manufactures products of different types in the same factory or production line. Normally, if many products need to be sorted into different boxes, this would manually be done by factory workers before the products are transported to the retailer. The Nirikshak Bot can do this automatically by using the ball balancing platforms to drop products into their designated boxes, right after they are manufactured, thus reducing manufacturing cost and time.

**Testing your knowledge (Theme and Rulebook analysis)**

**Q2. Consider the following dictionary written in ball\_details.json file:**

**{**

**“red” : [“T3\_CB1”],**

**“green” : [“T2\_CB2”, “T1\_CB1”],**

**“blue” : [“T1\_CB3”, “T3\_CB3”]**

**}**

**Based on the dictionary given above, write the correct Collection Box for the following sequence of balls dispensed by BD: (5)**

< This question is to check if you have understood how to interpret the ball\_details.json file correctly. Hence fill in the answers carefully in the table below>

|  |  |  |
| --- | --- | --- |
| **Sequence** | **Color** | **Collection Box Name** |
| 4th | Green | T1\_CB1 |
| 5th | Blue | T3\_CB3 |
| 2nd | Blue | T1\_CB3 |
| 3rd | Red | T3\_CB1 |
| 1st | Green | T2\_CB2 |

**Q3. Consider the JSON configuration given in Q2.**

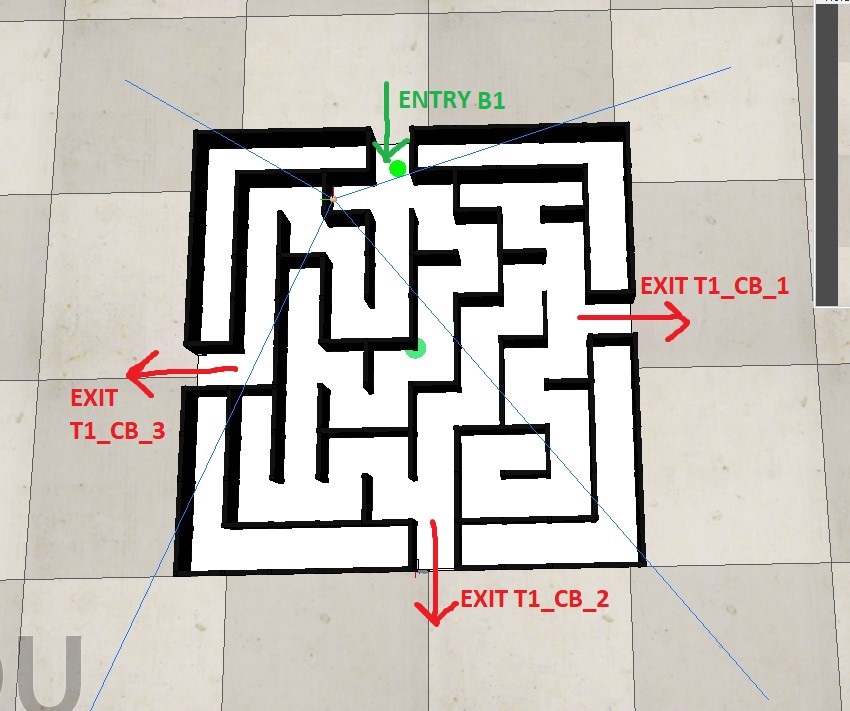
1. **What are the ENTRY and EXIT cell coordinates used by the *first green ball* for all the tables it is passing through? :** T4: Entry coordinates= (0,5) Exit coordinates= (9,4) T2: Entry coordinates= (0,4) Exit coordinates= (9,5) **(2)**
2. **What are the ENTRY and EXIT cell coordinates used by the *second blue ball* for all the tables it is passing through?:** T4: Entry coordinates= (0,5) Exit coordinates= (4,0) T3: Entry coordinates= (4,9) Exit coordinates= (0,4) **(2)**
3. **What are the ENTRY and EXIT cell coordinates used by the *first red ball* for all the tables it is passing through?** T4: Entry coordinates= (0,5) Exit coordinates= (4,0) T3: Entry coordinates= (4,9) Exit coordinates= (9,5) **(2)**

< This question is to check if you have understood Arena section of the Rulebook. Write your answers point wise for (a), (b) and (c)>

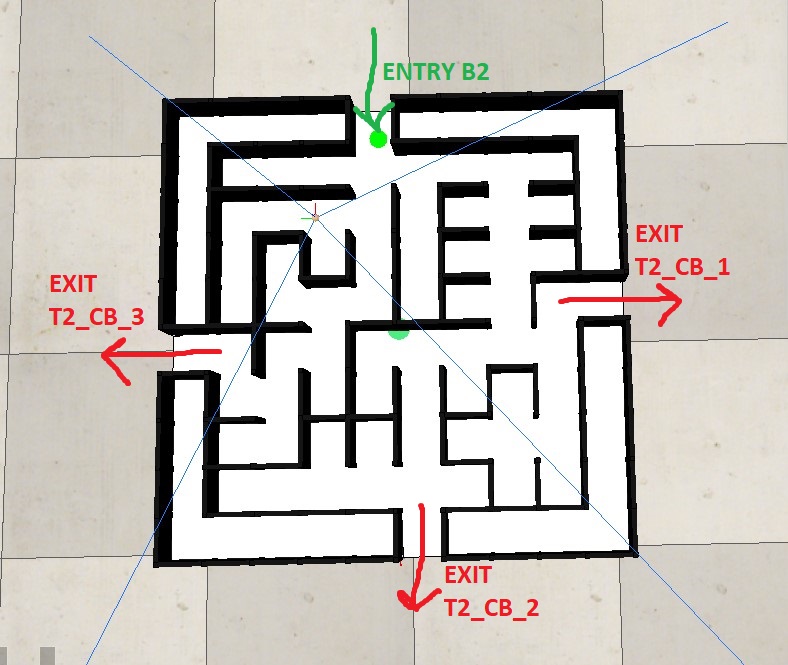
**Q4. Download the *task\_4c\_maze\_images.zip* file from this** [link](https://portal.e-yantra.org/storage/ryMPCVnKgx_nb/task4_files/task_4c/task_4c_maze_images.zip) **(from Task 4C page). The images have been named maze\_t1.jpg, maze\_t2.jpg and so on (according to the Theme Run Requirements part under Theme Description section of the Rulebook). Generate these mazes on the single Platform Table one by one according to the resultant maze images shown in Figure 10 and 12 of Arena section in CoppeliaSim and capture a top-view screenshot for all of them. (4)**

< Make sure to carve the respective EXIT points for all the mazes on Platform Table. Paste all the screenshots in this document. All the screenshot images should be properly labelled with ENTRY and EXIT clearly marked>

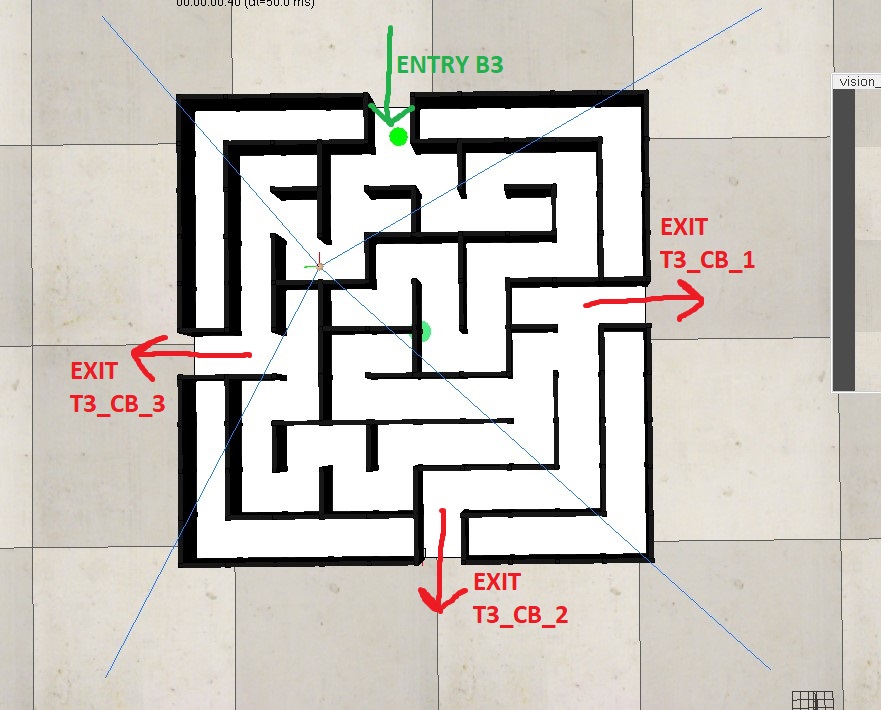
**MAZE T1:**

****

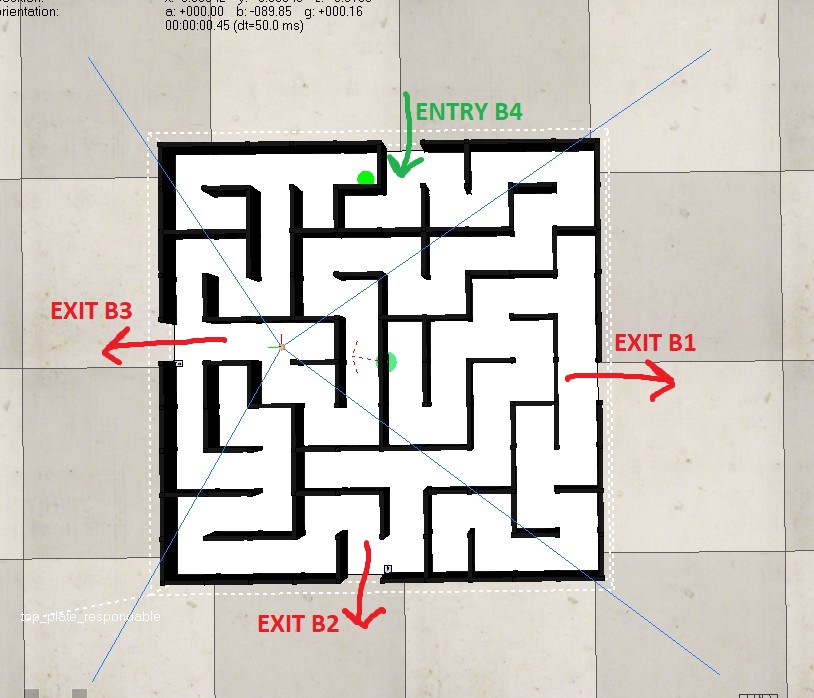
**MAZE T2:**

****

**MAZE T3:**

****

**MAZE T4:**

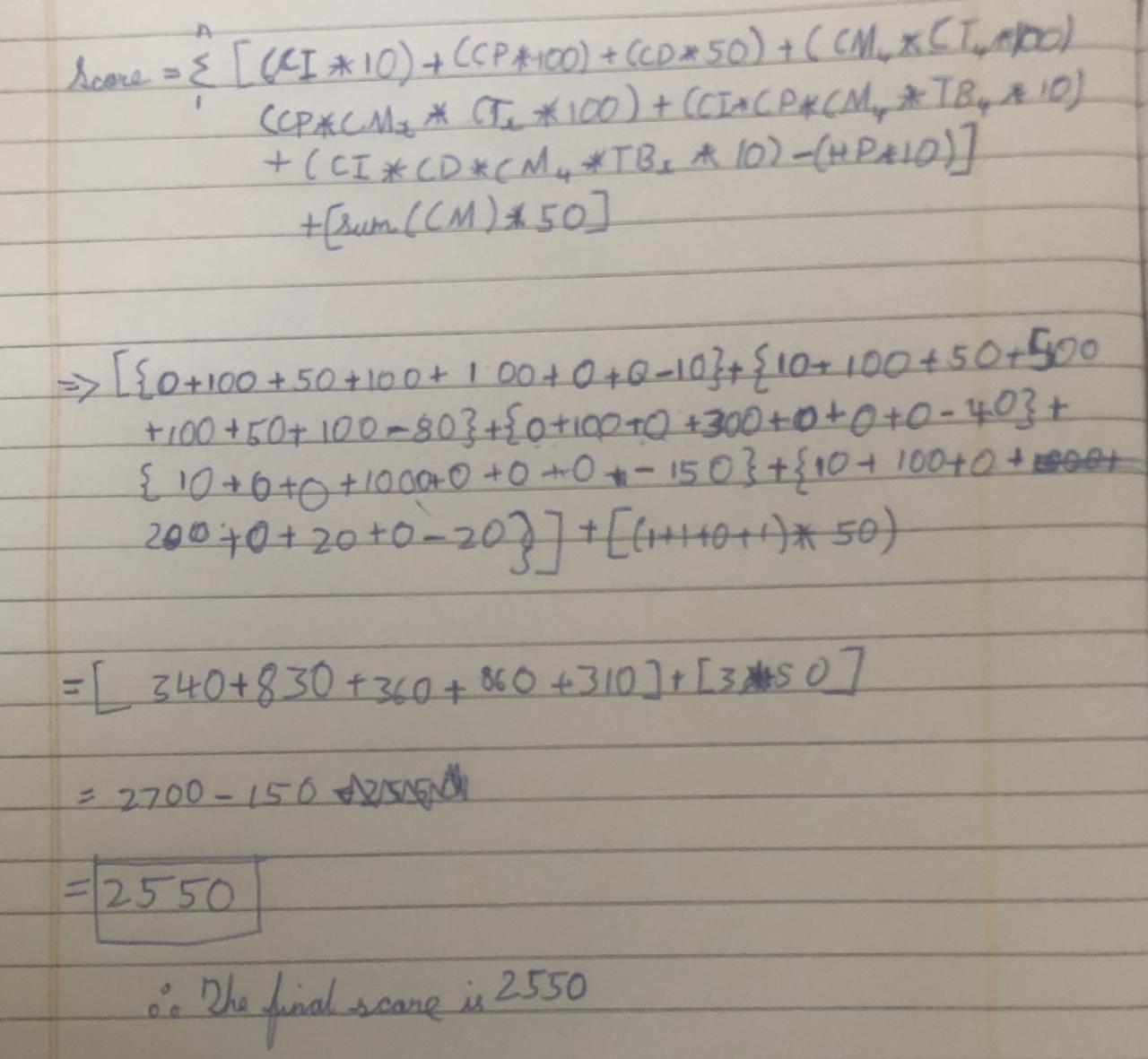
****

**Q5. Consider the following table showing the scenario for each ball and calculate the final score:**

**Final Score = 2550 (5)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **CI** | **CP** | **CD** | **CT4** | **CTx** | **TB4** | **TBx** | **HP** |
| **1st** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** |
| **2nd** | **1** | **1** | **1** | **1** | **1** | **5** | **10** | **8** |
| **3rd** | **0** | **1** | **0** | **1** | **1** | **3** | **13** | **4** |
| **4th** | **1** | **0** | **0** | **1** | **1** | **10** | **20** | **15** |
| **5th** | **1** | **1** | **0** | **1** | **1** | **2** | **17** | **2** |

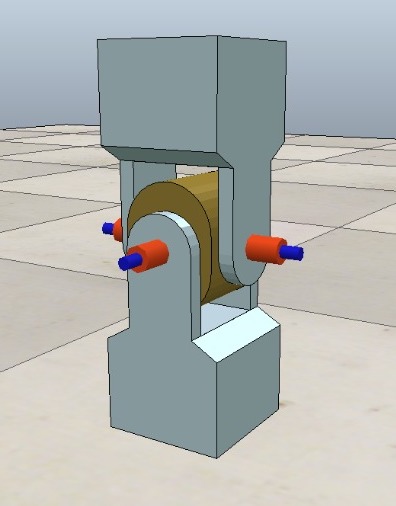
|  |  |  |  |
| --- | --- | --- | --- |
| **CM1** | **CM2** | **CM3** | **CM4** |
| **1** | **1** | **0** | **1** |

< Show your calculations in detail below>

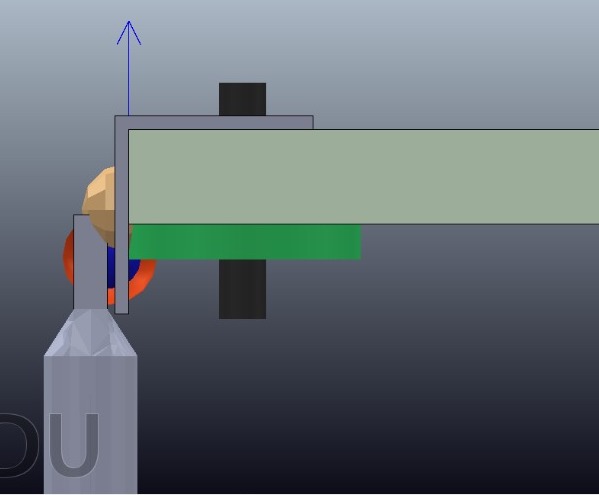
**Mechanism**

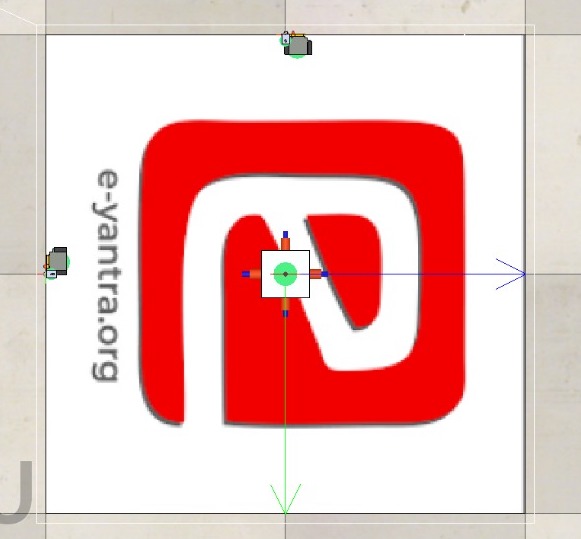
**Q6. Explain the mechanism that you used for your ball balancing platform. (5)**

< You must explain the mechanical construction of your ball balancing platform and how have you connected all the different components provided to you. Make properly labelled diagrams to show the same. You may also use screenshots of the CoppeliaSim scene to demonstrate your mechanism.>

1. The mechanism used in our ball balancing platform consists of two manipulator assemblies which tilts the top plate in a given direction.
2. The top plate is connected to the base plate with a universal joint assembly that allows the top plate to tilt about both the x and y axes.
3. The universal joint assembly consists of two yokes and a universal joint pinion with each yoke having an independent revolute joint with the universal joint pinion. One of the yokes is rigidly connected to the base plate while the other is rigidly connected to the top plate.
4. The arm assembly consists of a connecting rod and and l connector joint to each other with a spherical joint. The connecting rod is also connected to the servo fin with a revolute joint while the l connector is connected to the top plate rigidly.





1. The two manipulator assemblies are mounted along two perpendicular edges of the base plate as shown in the picture.

**Q7. In Task 1C, you were given the task to design the ball balancing platform while in Task 3, you were given the task to use this ball balancing platform to control the position of the ball on top of it. How did your ball balancing platform change between these tasks? (5)**

< Explain in brief how your design changed in the subsequent tasks. If your design did not change, then justify your reasons for the same.

Answer format: Text - limit: 100 words. .>

1. In the first design there were four manipulator assemblies, one along each edge of the base plate. This was changed to there being only two manipulator assemblies mounted along two perpendicular edges of the base plate to simplify the operation of the mechanism and to eliminate its conflicting elements.
2. The joint between the connecting rod and the l connector was changed from it being a revolute joint to a spherical joint to facilitate freedom of motion needed to tilt the platform in the required direction.

**Path Planning**

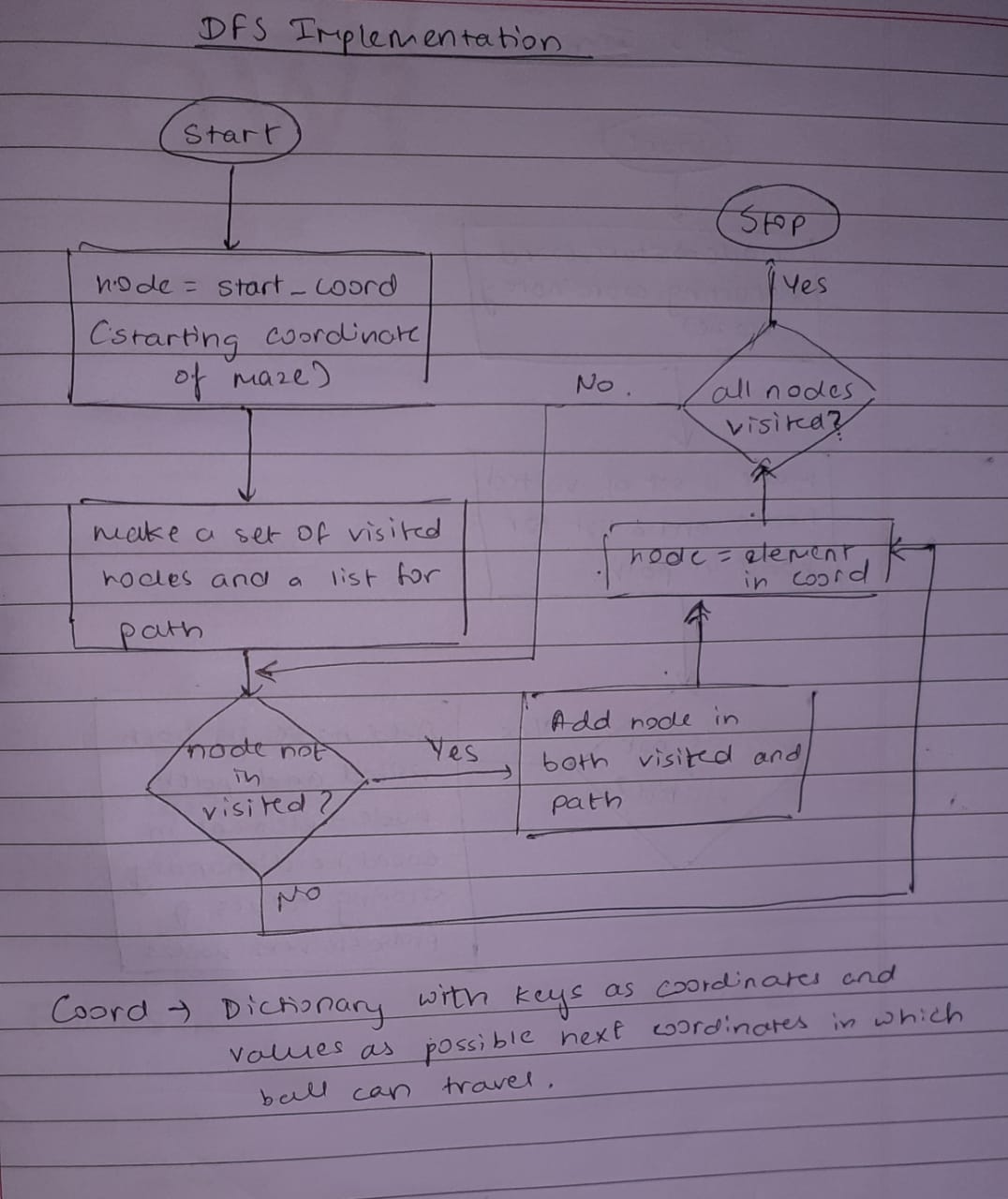
**Q8. What kind of path planning algorithm did you use for finding the shortest path for the given maze images (in Task 4A)? (5)**

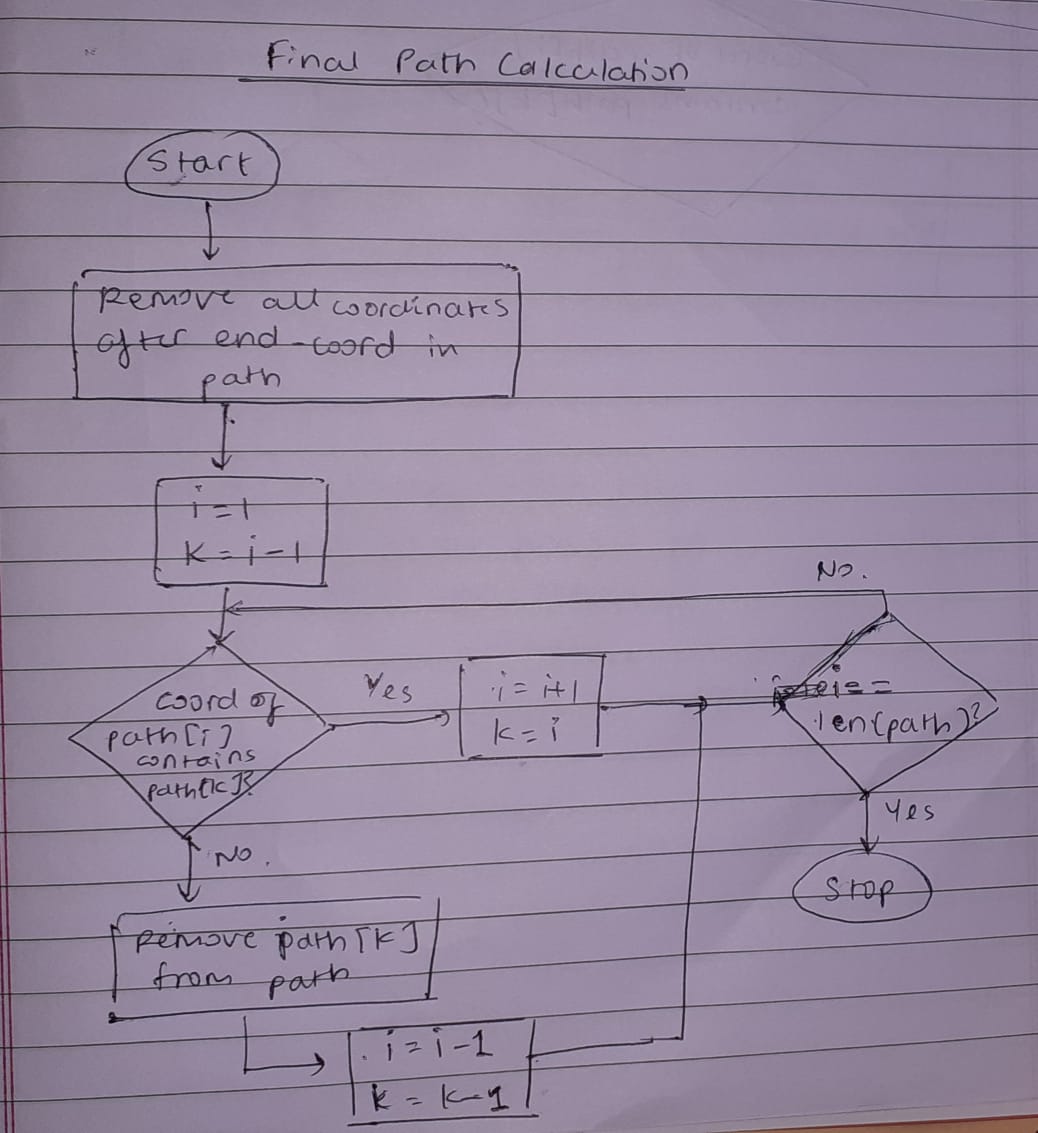
< Explain the logic behind the algorithm and the reason for your choice if any. You can use a pseudo-code and/or flowcharts to help elucidate your answer. >

We looked into several path planning algorithms such as BFS, DFS, Dijkstra’s, A\* etc. We realized that algorithms such as Dijkstra’s and A\* are very effective in path planning. However, theses algorithms require some sort of heuristics (cost of path from a particular node to final node) that are obtained from approximation methods or past experiences.

Our first attempt was to implement an algorithm similar to A\* which dynamically calculates the path by comparing the Euclidean distance of every possible node to the final node, and the node which has the least distance is chosen. This method is repeated for every node. The algorithm turned out to be very tedious to write and not very effective since the node with the least Euclidean distance was more than often not the node required for the correct path.

We thus discarded this algorithm and thought of implementing a simpler but effective algorithm like BFS or DFS. We studied both these algorithms and thought DFS was the best fit since it was faster and practically made sense since the approach used in DFS is similar to how humans think while solving a maze. A path is chosen and it is completely discovered. If this path is unsuccessful, the search is suspended and we explore a different path.



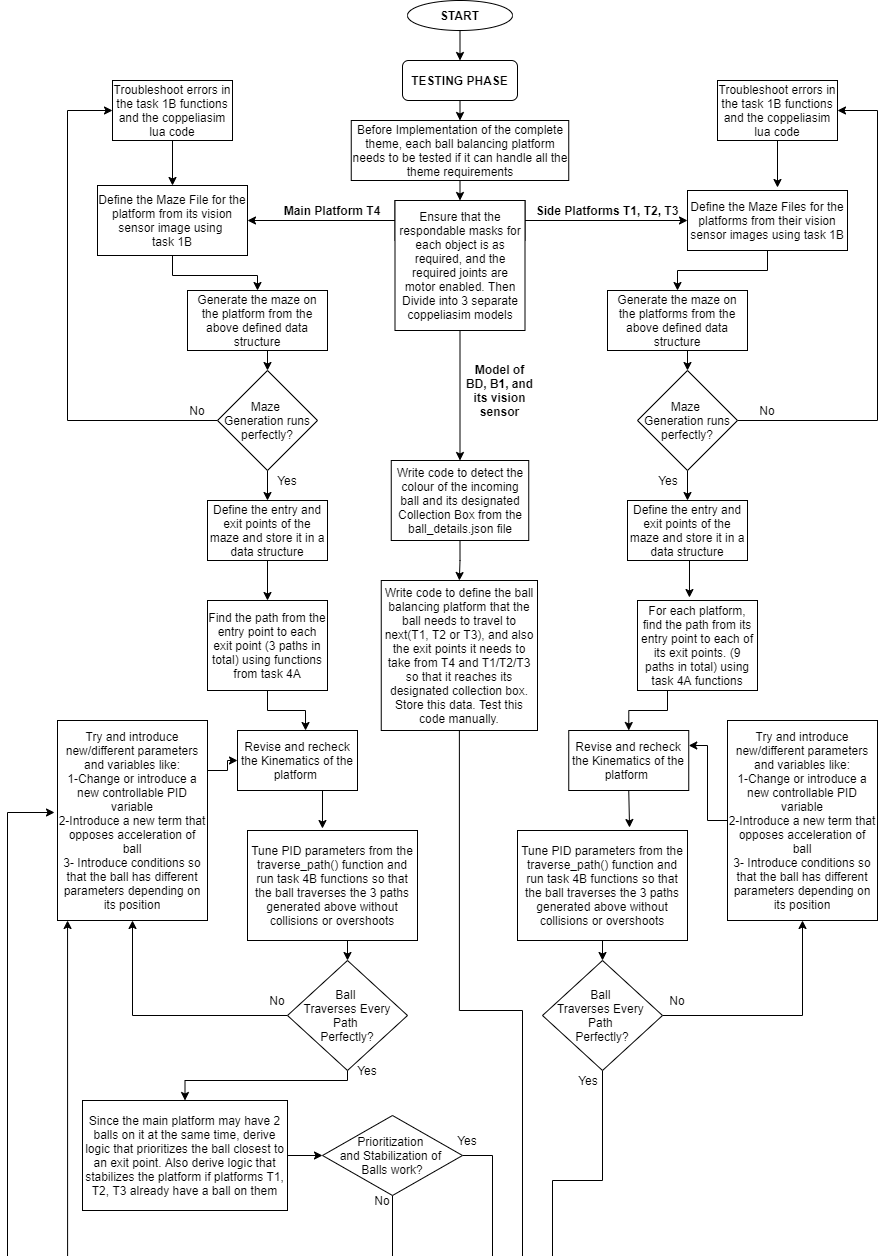


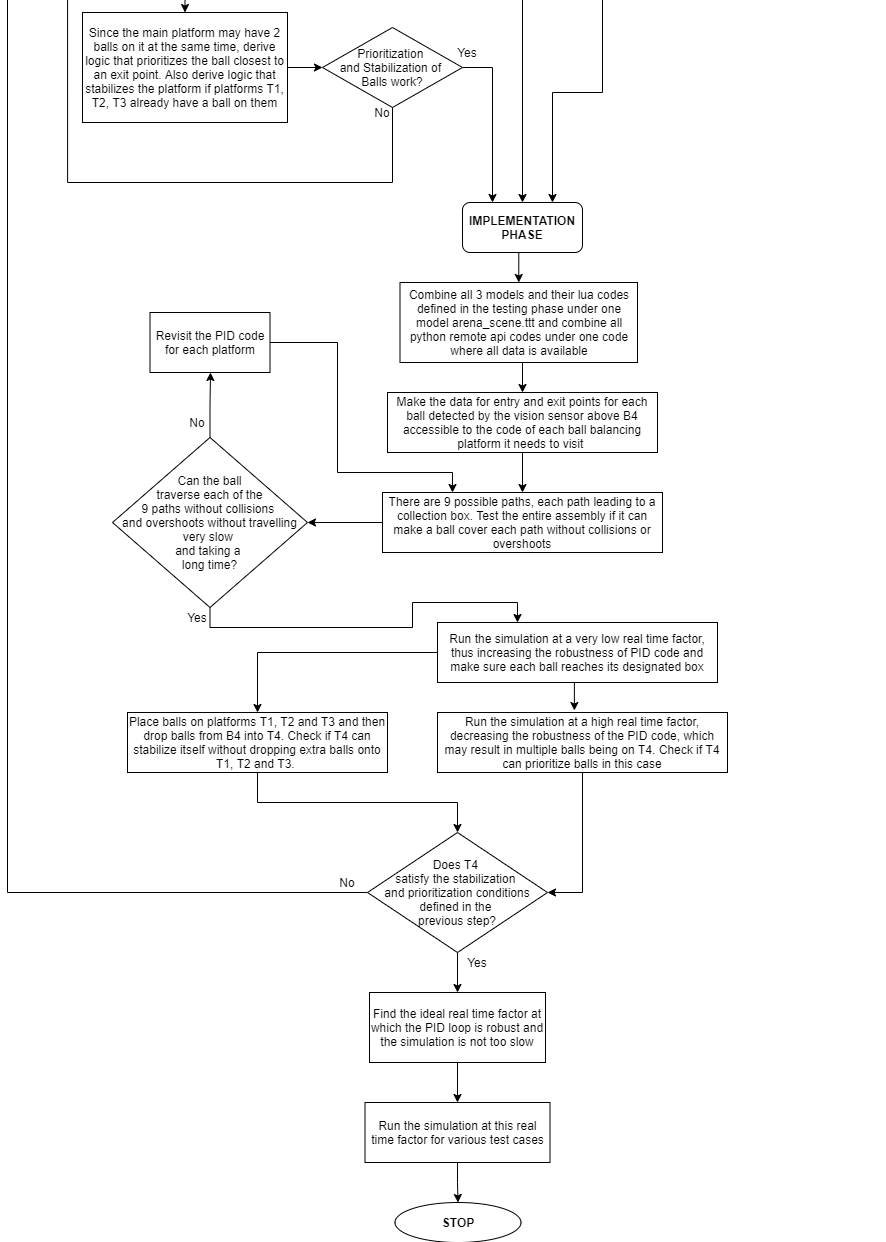
**Algorithm Analysis**

**Q9. Draw a flowchart illustrating the algorithm / strategy you propose to use for theme implementation. (7)**

< The flowchart should elaborate on every possible function that you will be using for completing all the Theme Run.

Follow the standard pictorial representation used to draw the flowchart. >



****

**Challenges**

**Q10. What are the major challenges that you have faced till now and the ones that you can anticipate in addressing this theme and how do you propose to tackle them? (3)**

< Answer format: Bullet points

1. Challenge 1

2. Challenge 2

3. Challenge 3, etc. >

1. Tuning the PID Loop for task 3, the dynamics of the ball balancing platform is very non-uniform in general, since the weight of the ball changes the system based on where it is, the ball moves slower when nearing the middle and moves faster when it is nearing an edge, in which case the differential term cannot keep up with the speed of the ball . Our design itself does not have a linear relationship between the angle of the servo and the angle of the top plate, which is a compromise we had to make for stability and independent control of each servo. This can be tackled by introducing code for forward and inverse kinematics to linearize the function.
2. The applyPerspectiveTransform() function is almost applied in all the tasks and used in different functions, it can act differently for different places its used in and can throw errors. This can be solved by fine tuning it or using different functions for different parts of the code.
3. Due to the limits of computational power, the PID Loop cannot be as robust as it should be in a real-life application, which leads to frequent overshoots of the ball because of the time complexity of each iteration. This forces us to reduce the stiffness of the PID Loop so that the ball travels slower and the code has enough time to run without the ball straying away, which may cause the simulation to exceed the time limit. This can be limited by running the simulation with as low a real time factor as possible, so that we can increase stiffness and make the ball move faster, therefore making the code more robust.

**IMPORTANT:**

**- The document you submit should be in YOUR OWN WORDS. To avoid any copyright violations, you must NOT copy phrases directly from manuals or the web.**

**- The team should NOT mail or upload the document anywhere else except on the portal.**

**- Teams failing to submit the document by the deadline will lose the marks for this task.**

**- e-Yantra WILL NOT entertain any request for extension of deadline for uploading the task.**

**- e-Yantra holds complete discretion to disqualify a team if any foul play is suspected.**