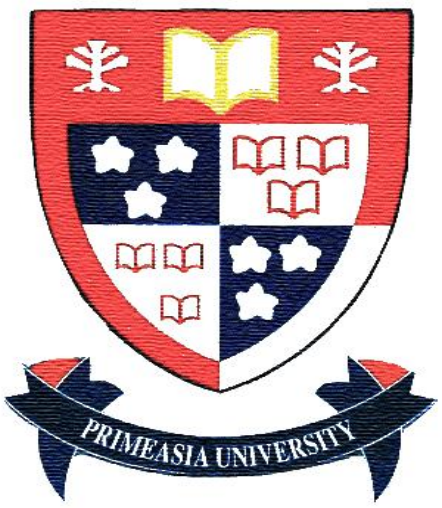
"Finding an/the easiest way to solve a 3x3x3 Rubik's cube and implementing it to build an Raspberry pi based solver machine using 5 stepper motors."

Submitted
In partial fulfillment
For the award of the Degree of
Bachelor of Science
in Department of Computer Science and Engineering



Primeasia University

Prepared by:

Shakur Mahmood (191008042)
Md Nasir Uddin (171005042)
Shahadat Khan Sakil (171025042)

Supervised by:

Farhin Farhad Riya
Lecturer, Department of CSE.

September 2021

Candidate's Declaration

We hereby declare that the work, which is begin presented in the project report, entitled **"Finding an/the easiest way to solve a 3x3x3 Rubik's cube and implementing it to build an Raspberry pi based solver machine using 5 stepper motors."** in partial fulfillment for the award of Degree of "Bachelor of Science" in department of computer science and engineering, Primeasia university is record of our own investigations carried under the guidelines of Farhin Farhad Riya, department of computer science and engineering, Primeasia university.

We have not submitted the matter presented in this project report anywhere for the award of any other degree.

Shakur Mahmood

(191008042)

Md Nasir Uddin

(171005042)

Shahadat Khan Sakil

(171025042)

Approval

Final year project has been accepted in partial fulfillment of the requirements for the degree of BSc in Computer Science & Engineering (CSE)

Project Supervisor

……

Farhin Farhad Riya
Lecturer, Department of CSE
Primeasia University

Project Examiner

……..

Mustafa Hasan
Assistant Professor & Head, Department of CSE
Primeasia University

Table of Contents

Topic……Page no.

Acknowledgement

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. We are very grateful to our project supervisor Farhin Farhad Riya for the guidance, inspiration, and constructive suggestions that help us in the preparation of this project.

Abstract

Rubik's cube solving specially a 3x3x3 Rubik's cube is an intelligent game since it was invented in 1974. It seems like a small particle which we can grab in our one hand but it is not that easy that some people think. Why? Because even winning a 3x3 tic-tac-toe game needs some tricks to follow and in case of Rubik's cube there are 6 tic-tac-toe sides and each side's each small cube is not only interrelated with its own side's other cubes but also interrelated with all the other sides every single cube. The fact here is, the researchers in this area are mainly young and they prefer to solve the Rubik's cube in a fast way rather than solving it in an energy efficient way. As a result, they need to learn or memorize more algorithms. In worst case more than 100 algorithms. It is really difficult for general people who only want to solve the cube in an energy efficient way rather than speed solving and this area should get attention by some talent gamers or, game researchers also after completion of our research.

Chapter One:
Introduction

Before the 3x3x3 Rubik's cube was invented the 2x2x2 Rubik's cube was both invented and Larry D. Nichols got patent in 1972. Then in 1975 Arno Rubik got patent for his 3x3x3 'Magic Cube'. It took full 1 month for him to solve the first scrambled cube. He made this cube intentionally but he observed that it is difficult to rearrange them to their original position. Since that time many people tried to find a way to solve the cube efficiently. Then people noticed that using algorithm makes it easy to solve the cube. People also noticed that doing almost all the algorithm repeatedly for 6 times take the cube to the starting point from where the algorithm started. Since then, people were searching for the algorithm, by doing which repeatedly will solve the Rubik's cube. So, we can understand that this algorithm needs to traverse all the possible 43 quintillion (43,000,000,000,000,000,000) scrambled configurations. And for this complexity this algorithm is called 'Devil's algorithm'. Still at present devil's algorithm is not invented. Then researchers realized that from any scrambled position the least number of steps needed to solve the cube is a fact in Rubik's cube. This number is called 'God's number'. Google's some algorithms have shown us that this number is 20. Previously researchers thought this number will be in between 54 to 20.The main problems of not finding out the easiest way to solve a scrambled Rubik's cube are-

- ✧ First of all young children's wastage of time because some children and their parents think, trying to solve a Rubik's cube is good for children's increasing brain function. But it is not a matter of children to memorize all the 100+ algorithms to solve a Rubik's cube.

- ✧ Some people think it is not possible for them to solve the Rubik's cube. They remain in a great frustration.

When people will able to learn that there is at least an efficient way to solve the Rubik's cube they will feel relief and able to use their efforts in finding out another efficient way (if exists) but not memorizing algorithms. And we believe understanding a fact is an important fact for a nation.

1.1 Summery

A 3x3x3 rubik's cube solver can solve any scrambled position. The main fact here is how it will solve the cube or, the way it will go ahead. First of all the machine gathers information of the current condition of the cube and then the machine starts running according to the instructions in the code. In a word the code will decide which stepper motor will run in which moment as one of the main work of the solver machine is to run the stepper motor.

1.2 Goals of this project

The main aim of this project is to find an easy, effective and easy to remember way to solve any scrambled rubik's cube and implementing it in machine using stepper motor and if possible the project will reduce the number of stepper motors needed. Speed cubing is little bit different than easy solving, here number of rotation can be higher than the speed solving. And for implementing this way in machine we have used python as our programming language because our group members have experiences in writing tic-tac-toe game code using c, c++, java and python and except python all other languages were needed 1200+ lines, where in python we were needed to write <500 lines of code. Moreover, we have learnt that python is the only code which we can run using arm based computer like raspberry pi. We could use Arduino language but it is not that easy as python. And we have also learnt that python is getting more popular nowadays in every section of computing. So, we will also able to learn many new things on python during this large project.

1.3 Getting started

If we want to know how a rubik's cube solver machine works we will notice only one type of machine is working actually, that is stepper motor. In our project we have used Nema 23 stepper motors which have high torque to rotate the sides of the rubik's cube. Behind the scene can be a lot of things because there must be a programmable electric way which will run those stepper motors. In a word the code allows the current to flow in right stepper motor in right direction as stepper motor runs with electromagnet. And how to solve the cube is generally written previously in that program. In addition to this all the rubik's cube solver machines have an input system where it gathers the initial color of each side's each small cube. It can be an automated color recognition system or, manual input system. To reduce the percentage of errors in our project we have used manual input system.

1.4 Features of a Rubik's Cube Solver machine

- Input System (it can be either automatic or, manual) for knowing the color of small cubes of each side.
- Program to solve the cube from that position (or, any other positions).
- Program to run the stepper motors according to solving way.

1.5 Target and objectives

i. Main Target

1. Finding an easy to solve any scrambled cube.
2. Implementing it in machine.
3. Trying to reduce the number of stepper motors needed.

ii. User's understanding

Our important target is to teach the user of our solving machine how to solve any scrambled cube in an easy way.

iii. Saving user's time

Our newly invented way should save users valuable time to learn how to solve a rubik's cube.

iv. Reducing beginner's frustration

Using our newly invented way users who are in the beginning stage will find that at least there is an easy and effective way to solve the rubik's cube. So, they should not fall in frustration.

v. Easy algorithms

While there is needed to use algorithms we need to make the algorithms similar or, nearly similar so that the learners do not face any difficulty while learning.

Chapter two
Literature Review

3x3x3 Rubik's cube is the most popular toy in the world till present. This is most probably because of it's multi color combination, tricky to solve, easy to grab and some people's thinking it is not possible for them to solve a scrambled cube give some people interest to solve the cube. Most of the cubers solve the cube memorizing algorithms. In this case the steps are generally making white cross matching with the center pieces of all the sides, solving all 4 top corners, solving second layer edges, making bottom cross, then OLL (Orienting Last Layer) & PLL (Permutation of Last Layer). But speed cube solvers memorize or follow some more algorithms. As a result they can solve the cube under 10 seconds. But obviously they are at professional level and have spent a lot of time. As mentioned before speed cubers are generally young and they prefer to solve the cube at a high speed rather than spending time to find an easiest way. Moreover, there are a lot of competitions around the world for speed cubing which make more interest to them. The Rubik's Cube Solver Machines around the world also teach the users or, follow speed cubing algorithms. There are some works on finding easy way to solve a scrambled cube like the algorithm called BROLL which is the easiest and effective way to solve OLL but we didn't find any effective way in combined to solve the whole rubik's cube in the easiest way. From this lacking our research got idea to find an/the easiest way to solve any scrambled cube. Here users need not to memorize a lot of algorithms but need to follow some instructions that we have found during our research.

2.1 Procedures of the project

As mentioned before that our instructions will make the 3x3x3 rubik's cube solving much easier. These instructions are-

- After solving the white cross or, top layer cross if we solve the second layer's edges will make the rubik's cube solving much easier. Then we will solve the top layer's 4 corners, then we will make bottom cross and at last we will follow BROLL and PLL.
- While solving the top cross or, middle edges when we will find the matching part we will put that to the bottom layer (if we do not find it in the bottom layer), which makes it much easier to make the top cross or, to solve the middle edges.
- To solve the top corners without ruin the middle edges we will use our first algorithm. It has 3 rules but similar algorithms.

- Then we will use the algorithm to make bottom cross but in this case we need not to match the edge parts.
- Then we will use BROLL algorithm to solve the bottom side (not the whole bottom layer).
- At last we will use PLL algorithm to solve the whole last layer.

One of the main significance of our project is while solving the rubik's cube we need not to rotate the top layer at all and top layer will always stay on top.

In the code section we have followed the above steps to solve the scrambled cube after gathering the information of the current position of the cube. The only fact we were needed to do was to write the code for every possible situation. For example, to solve one piece of top corner we were needed to write same algorithm for 16 times. But out of 16 possible cases only one case will work. So to solve all the four corners we were needed to write the same code for 64 possible cases but only 4 case will work.

For the machine part we have used the same code. We have saved our output in an array and for every element of that array we have set a stepper motor to rotate either clockwise or, anticlockwise. For example, if the element is 'F' front side stepper motor will rotate clockwise and if the element is 'f' front side stepper motor will rotate anticlockwise. We have used an arm based computer named 'raspberry pi' to control the stepper motors. Also we have used 5 pieces of drv8825 drivers to control the current flow in the stepper motors.

Our stepper motors were 6 wired. So, we were needed to identify the 2 neutral wires. It is an easy work. We were needed to measure the wire resistances using a multi-meter The 3 wires which are from the same coil will show reading on the multi-meter. The middle valued wire is a neutral wire. For the remaining 3 wires we were needed to do the same test to identify the other neutral wire. In most of the cases yellow and white colored wires are the neutral wires but to identify the wires from the same coil we need to test it.

The pick current limit of the drv8825 driver is 1.5 ampere and in this driver current flow rate is double the voltage we set. But to reduce anxiety we have set the voltage to 0.5 volt.

2.2 Summery of Literature Review

- Instead of only memorizing algorithms our project also emphasizes on following some instructions.
- One of the main significance of this research is top layer need not to rotate at all and top layer will always stay on top. So, we can understand that we need to use 5 stepper motors to rotate the other 5 sides. Thus we were able to lower down the number of stepper motors needed to solve the rubik's cube.
- All the algorithms are actually similar with each other.
- Writing the code took a lot of effort but without considering all the possible cases we can not think of correct output at all.
- 'Raspberry pi' was used to control the stepper motors.
- 'drv8825' drivers was used to control the current flow to the stepper motors.
- We were needed to flow correct amount of current in the stepper motors.
- We were needed to identify the wires from the same coil. Our stepper motors have 2 phase and there are 2 coil pairs and 8 poles.

Chapter Three
Methodology

A software development methodology is a framework that is used to structure, plan, and control the process of developing an information system, this includes the pre-definition of specific deliverables and artifacts that are created and completed by a project team to develop or maintain an application. A wide variety of such frameworks have been produced over the years, each with its own recognized power and weakness. One software development methodology framework is not necessarily suitable for use by all projects. Each of the available methodology frameworks is best suited to specific kinds of projects, based on various technical, organizational, project, and team considerations. These software development frameworks are often bound to some kind of organization, which further develops, supports the use, and promotes the methodology framework. The methodology framework is often defined in some kind of formal documentation. Specific software development methodology frameworks include:

✓ Rational Unified Process (RUP, IBM) since 1998.
✓ Agile Unified Process (AUP) since 2005 by Scott Amber

Every software development methodology approach acts as a basis for applying specific frameworks to develop and maintain the system. Several system development approaches have been used since the origin of information technology. We have used the waterfall model in our project. Now we are going to discuss this system below.

3.1 Review on waterfall model

The Waterfall Model is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. This type of model is basically used for a project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In this model the testing starts only after the development is complete. In the waterfall model phases do not overlap.

3.1.1 Advantages of waterfall model

- This model is simple and easy to understand and use.
- It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- In this model, phases are processed and completed one at a time. Phases do not overlap.
- The waterfall model works well for smaller projects where requirements are very well understood.

General Overview of Waterfall Model

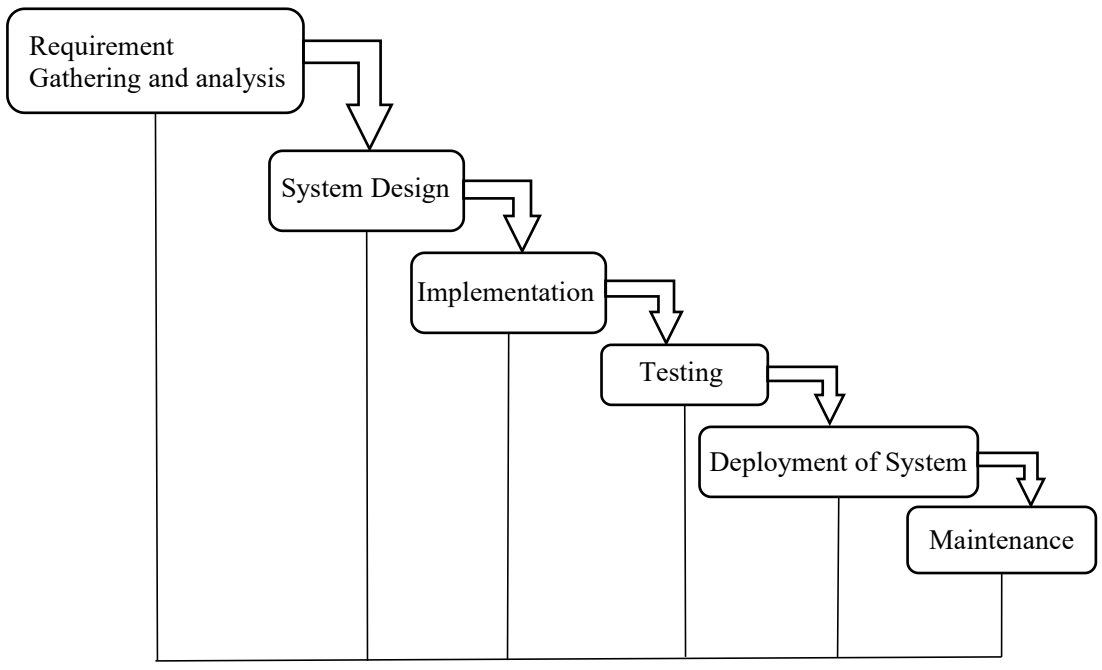


Figure 3.1: Waterfall model of the project

3.1.2 Disadvantages of waterfall model:

• Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought-out in the concept stage.
• No working software is produced until late during the life cycle.
• High amounts of risk and uncertainty.
• Not a good model for complex and object-oriented projects.
• Poor model for long and ongoing projects.
• Not suitable for the projects where requirements are at a moderate to high risk of changing.

3.1.3 When to use the waterfall model

- This model is used only when the requirements are very well known, clear and fixed.
- Product definition is permanent.
- Technology is understood.
- There are no indefinite requirements.
- Sufficient resources with the required expertise are available freely.
- The project is short.

However, very less customer enter action is involved during the development of the product. Once the product is ready then only it can be demoed to the end-users. Once the product is developed and if any failure occurs then the cost of fixing such issues is very high because we need to update everywhere from document to logic.

3.2 Justifying chosen methodology

To solve actual problems in an industry setting, a software engineer or a team of engineers must incorporate a development strategy that encloses the process, methods, and tools layers and generic phrases. This strategy is often referred to as a process model or a software engineering instance or project development approach. A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required. This project is based on Waterfall Model. The waterfall model is a simple software development process model that punctuates a long development period. If requirements are well understood and the project scope is awkward, the waterfall process enables a development team to create a “fully functional system” within short time periods. And for this System development which falls within a short period of time, there is no other methodology suitable other than the waterfall model, which is the best approach in producing the expected output for the Rubik’s Cube Solving Machine.

3.3 System Requirement Analysis

This includes the development environment and users environment where they will use this system. Our project requires the following materials-

1. Arm based computer like Raspberry pi 4 (4gb) or, its compatible device with good heat sink or, cooling fan.
2. Python 3.0 installed.
3. Python code development IDE.
4. 5 stepper motors with high torque like Nema 23 stepper motors.
5. 5 drv8825 stepper motor drivers.
6. 1 full sized breadboard.
7. A structure where we will put our stepper motors to rotate the sides of the rubik's cube.
8. A good quality rubik's cube which is light weight and rotates smoothly.
9. A small screw driver to adjust the current flow in the stepper motor driver.
10. Suitable power adapter according to the stepper motor's rated voltage and current (in our case 24 volt and 2.4 ampere power adapter). But best is to use digital adjustable power supply where we can adjust both the voltage and current flow rate.
11. 40 male to female and 60 male to male good quality jumper wires which can handle high voltage like 24 volt power supply.
12. And a good quality multi meter.

Chapter Four
System design and development

Our project's design is simple like the diagram below:

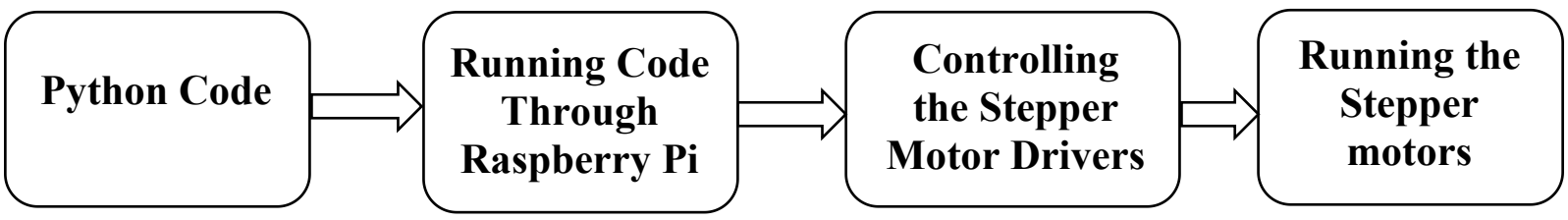
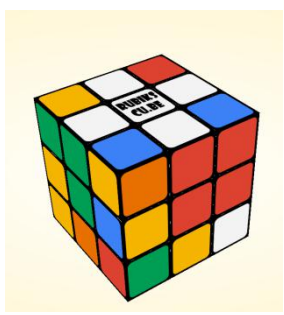


Diagram 4.1: Project design

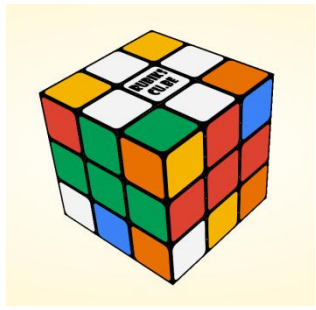
Writing the code took a lot of efforts. It took near 7500 lines of coding. In the code part we have used our newly found method to solve the rubik's cube. Now we want to discuss that method in detail-

4.1 The technique to solve a scrambled rubik's cube

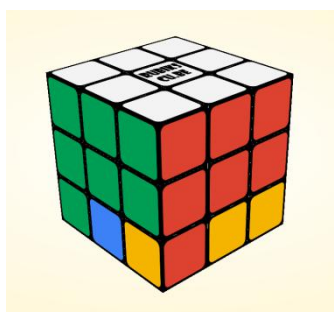
In brief -



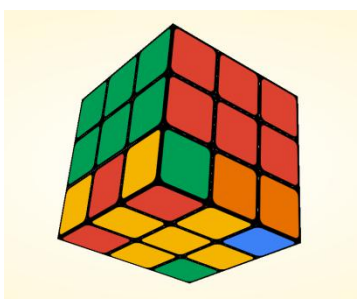
1. Solving White Cross



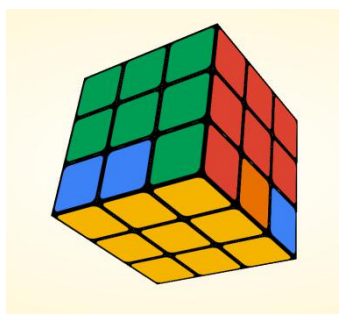
2. Solving 2nd layer's edge pieces



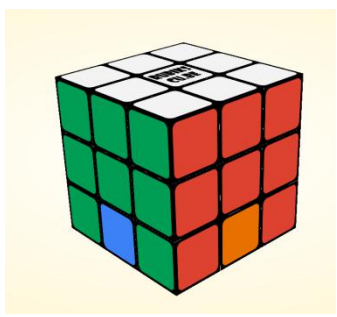
3. Solving the top corners



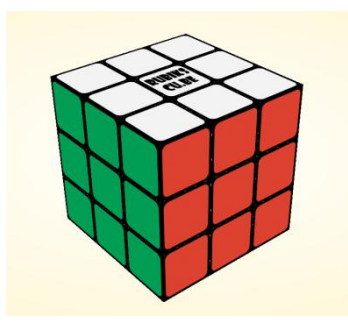
4. Making Bottom Cross



5. Solving the bottom side



6. Making headlights in all the sides



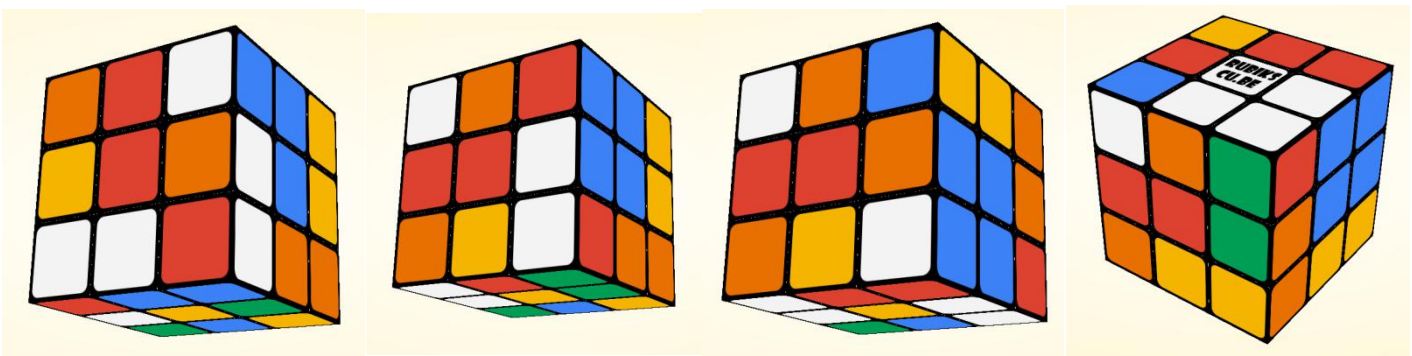
7. Solving the headlights

1. One of the significance of this technique is white color will face always on top and

2. We need not to rotate the top layer at all. So, we can make our solver machine using 5 stepper motors instead of 6.

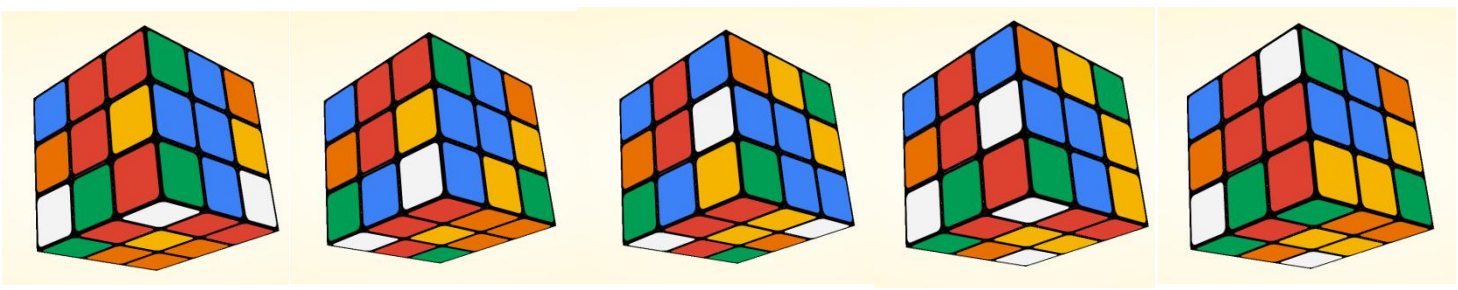
In detail-

First of all we will make top cross. To do this we need to find the top edge pieces containing white color and one other color that matches to the second layer's center pieces. If we do not find those in the bottom layer we will shift those to the bottom layer. Then if we find white color is not facing bottom side, we need to face it to the bottom side like the pictures below-



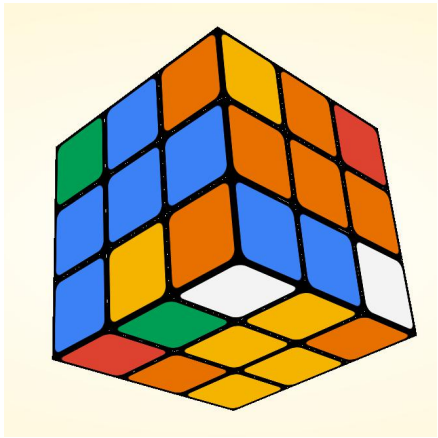
Here we can see white and blue colored edge is in the bottom layer but white color does not faced downward. So, we have rotated the front side anticlockwise once and than right side anticlockwise once. At last we have rotated right side clockwise twice and we have solved one piece of white cross.

After solving all the remaining white cross pieces we will solve the second layer's edge pieces with the same technique. We can provide an additional technique here that is after putting the required part in the bottom layer we can match the opposite color with the second layer's center pieces. Like the pictures below-

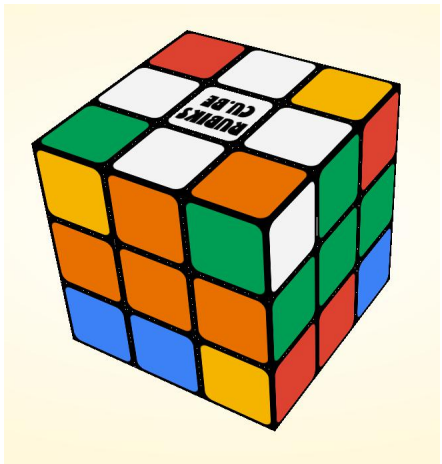


Here blue and red colored edge piece is required to solve the second layer. That piece is situated in the bottom layer with matching with blue color center piece. So, we have rotated bottom layer anticlockwise once so that red color is faced down side and blue color is aligning with the red color center piece. Then we should rotate the right side anticlockwise once so that our targeted position piece comes to the bottom layer. Now if we rotate the bottom layer clockwise once blue color will match with the second layer's center piece. At last rotating the right side clockwise once will solve the desired second layer edge part.

After solving all other second layer edges we will solve the top 4 corners. To do this we will use our first algorithm which will not ruin the second layer edges but it will solve the top corners. Here, after finding the desired top corner piece we need to put that to the bottom layer aligning to that top corner piece just like the picture below-



But if we find it in the top corners we need to take that to the bottom layer without ruin the second layer's solved part like the picture below-

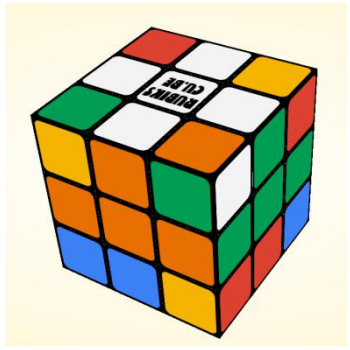


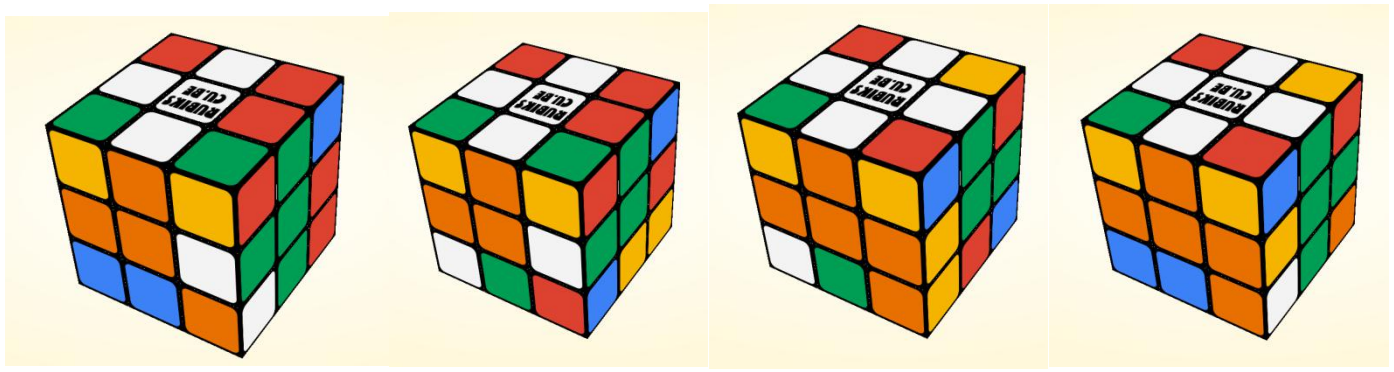
To do this we need to apply the algorithm thrice on that position. The algorithm is-

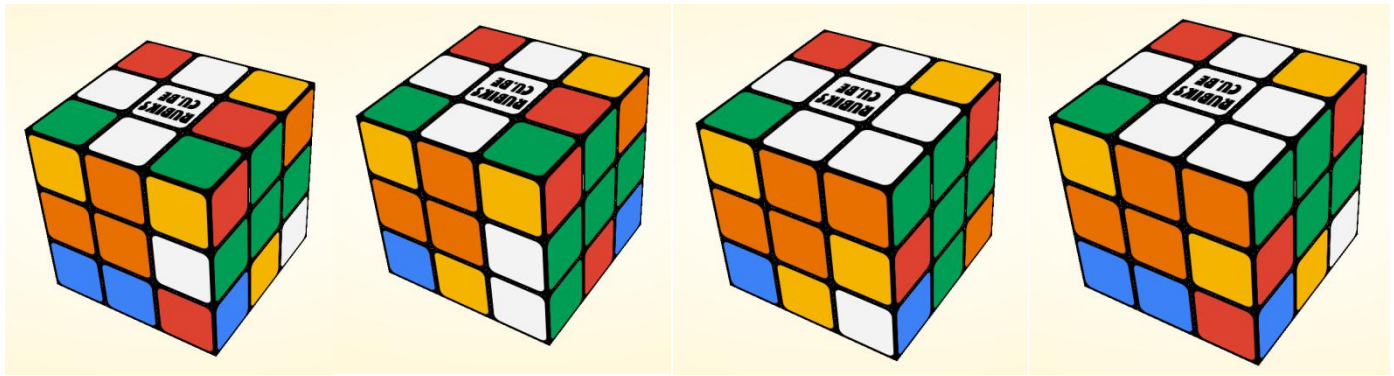
r d R D (right hand algorithm)
or,
L D l d (left hand algorithm)

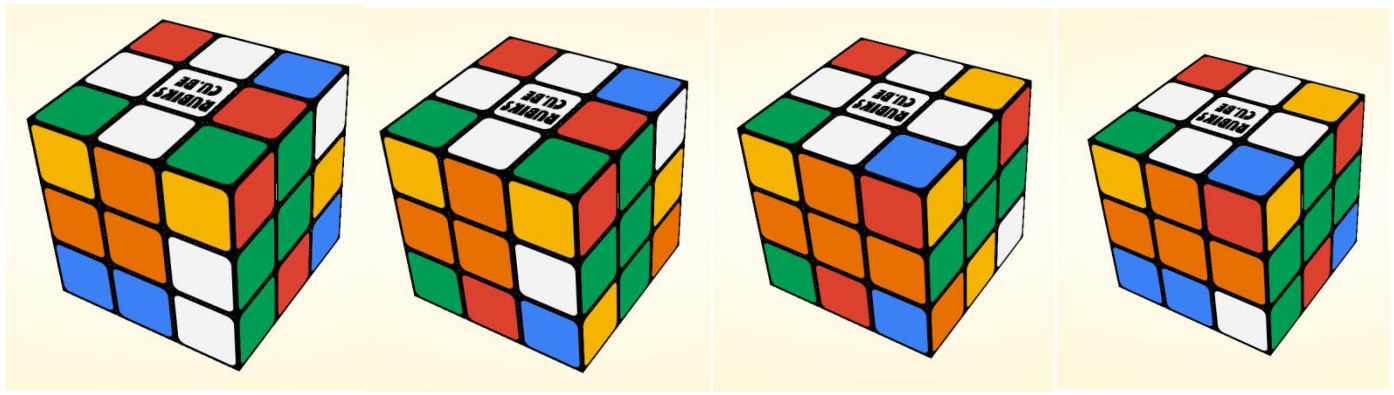
Here, capital letter is mentioning clockwise rotation and small letter is mentioning anticlockwise rotation.

In pictures-

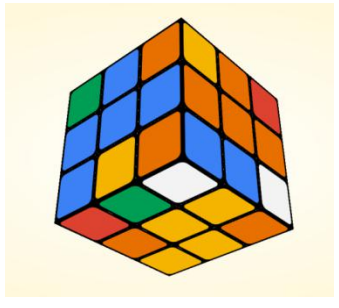


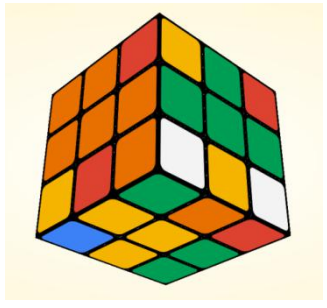
1. 

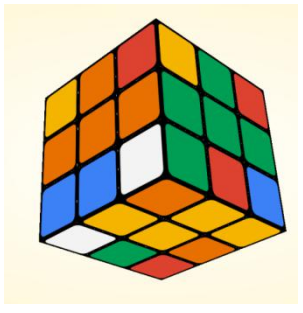
2. 

3. 

After putting that top corner piece we need to put that piece in the bottom part aligning with the desired top corner position. Then 3 possible case can be possible. First of all, white color is facing downside. Secondly, white color is facing right side. And lastly, white color is facing left side. These situations are likely the pictures below-

1. 

2. 

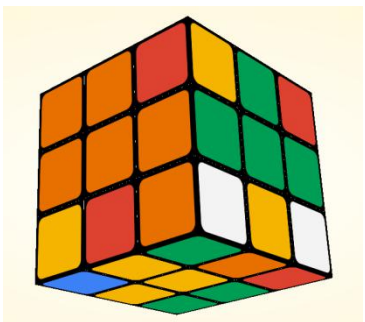
3. 

If white color is facing downside we should use our first algorithm thrice just like we have used it to put down the corners from the top layer to the bottom. But if white color is facing right or, left side we need to use that algorithm in slightly different way.

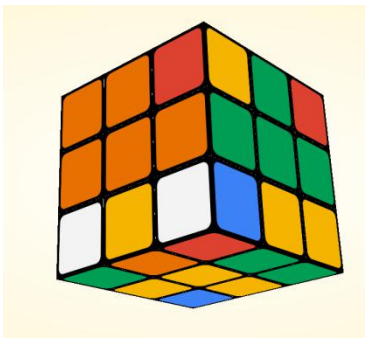
If white color is facing right side we need to rotate the bottom layer anticlockwise once and then we need to handle the rubik's cube in such a way so that, that white color is facing toward us. Then we need to apply right hand algorithm once. After that we need rotate the bottom layer clockwise once so that the white color corner piece gets its previous position again. Then if we apply right hand algorithm again on that piece will solve our one corner piece. So, in this case the combined algorithm is-

d (r d R D) D (r d R D)

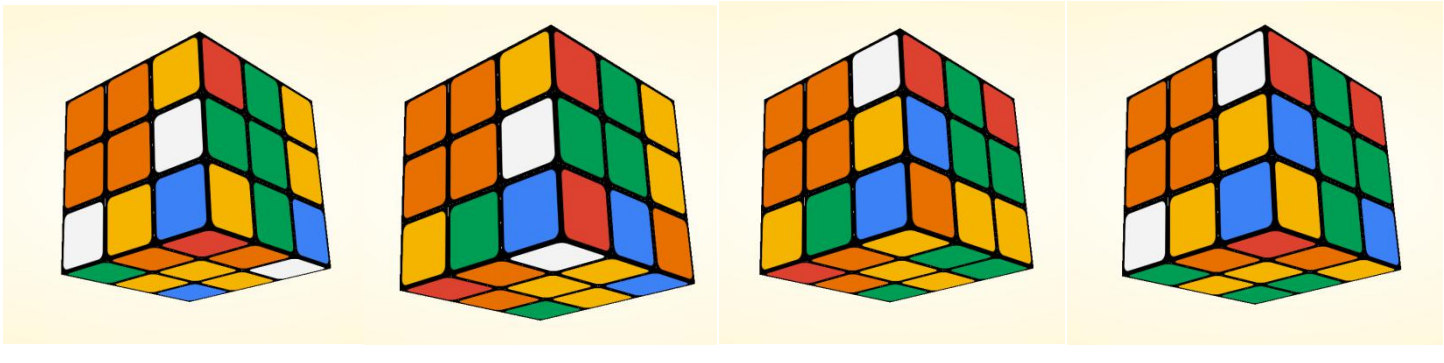
In pictures-



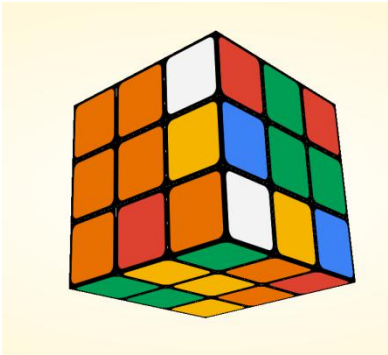
d



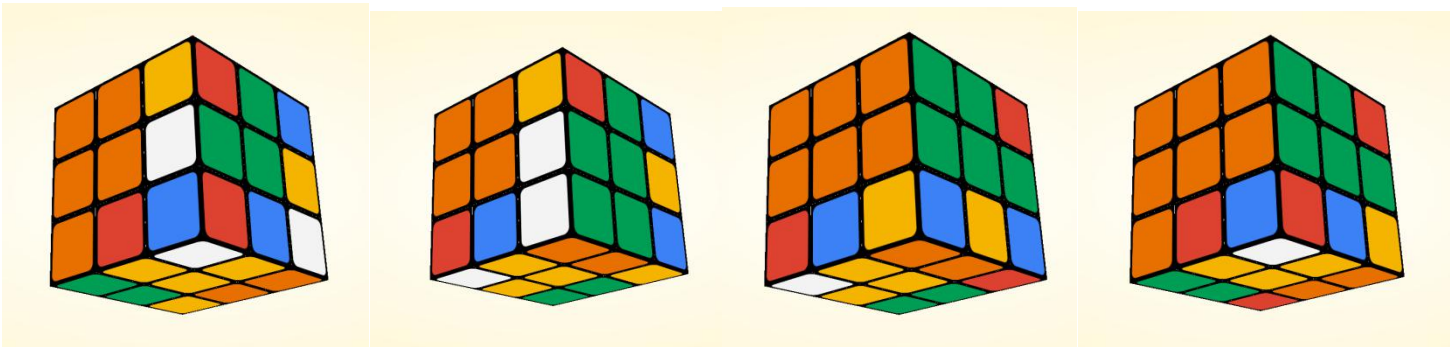
r d R D



D



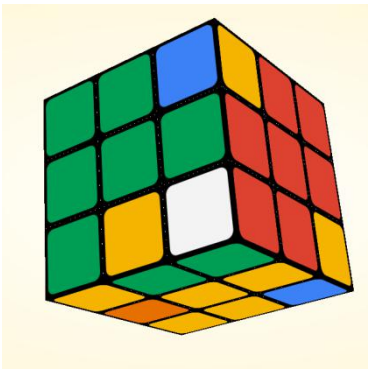
r d R D



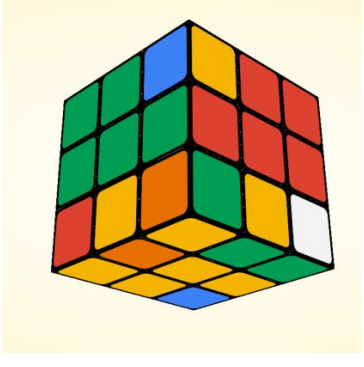
If white color is facing left side we need to rotate the bottom layer clockwise once and then we need to handle the rubik's cube in such a way so that, that white color is facing toward us. Then we need to apply left hand algorithm once. After that we need rotate the bottom layer anticlockwise once so that the white color corner piece gets its previous position again. Then if we apply left hand algorithm again on that piece will solve our one corner piece. So, in this case the combined algorithm is-

D (L D l d) d (L D l d)

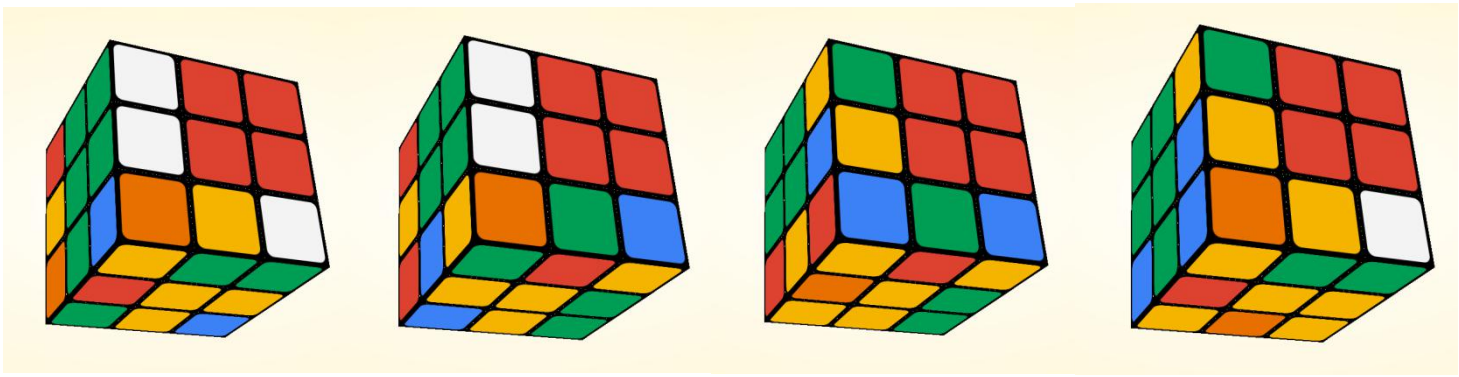
In pictures-



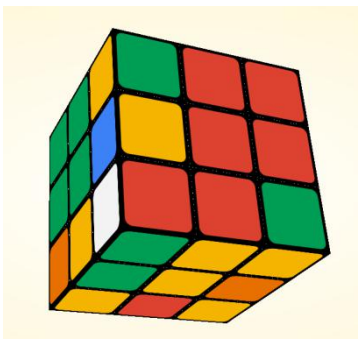
D



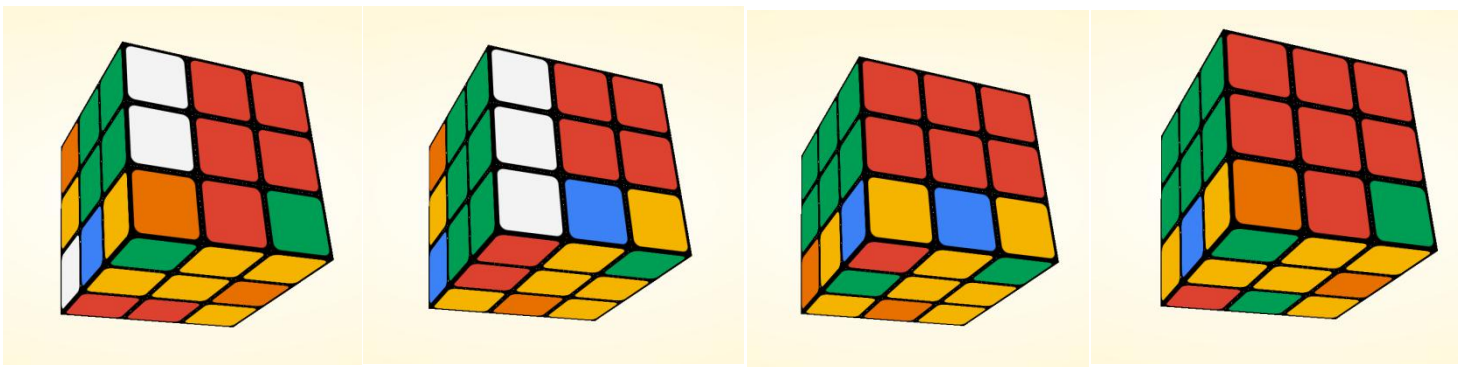
L D l d



d



L D l d



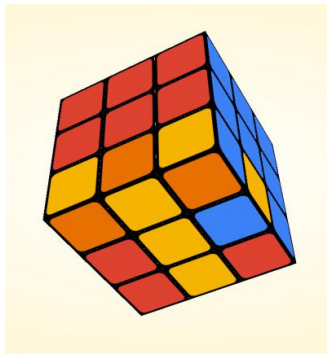
Using same process we can easily solve all the other top corners. So, actually using only one algorithm we have solved the top 2 layers.

Now, we will solve the bottom layer. To do this we will first make the bottom cross or, yellow cross but it is not mandatory to match the edge pieces with the 2nd layer's center pieces.

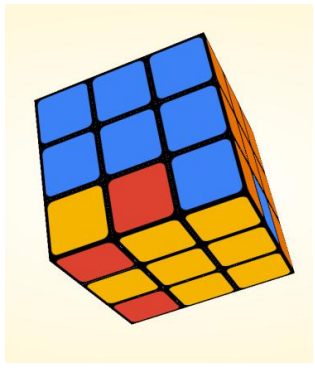
To make this bottom cross we need to use our 2nd algorithm which is-

f (r d R D) F (right hand algorithm)
or,
F (L D l d) f (left hand algorithm)

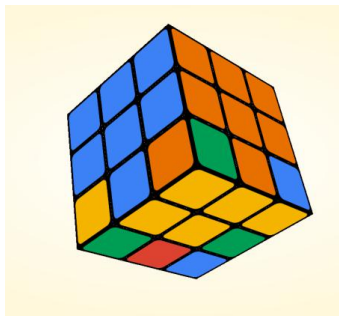
If bottom cross is not completed automatically, 3 situations can be possible. First situation is called 'the line'. Here, two edge pieces of opposite side are yellow and make a line with the bottom layer center piece like the picture below-



In this case we can either hold the cube in such a way so that the line remains in front of us horizontally or, we can rotate the bottom layer once to make it horizontal to us. Then if we apply second algorithm that will create our desired yellow cross which is showing in the picture below-

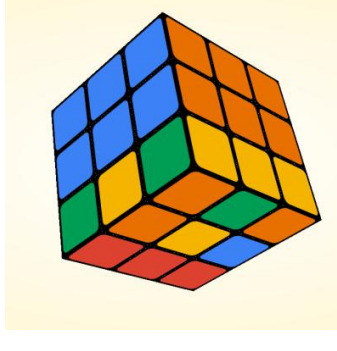


Second situation is called ‘L shape’. Here, yellow colored pieces make the shape like the letter ‘L’ with the center piece. It is like the picture below-



In this case we need to put that ‘L’ infront of us and need to apply the second algorithm which will create ‘a line’ and we have discussed before what to do with ‘a line’.

Third situation is called ‘dot situation’ where there is no yellow edges or, at maximum one yellow edges facing downward like the picture below-



If we apply second algorithm on this situation that will create the ‘L shape’ and now we know what to do with the ‘L shape’.

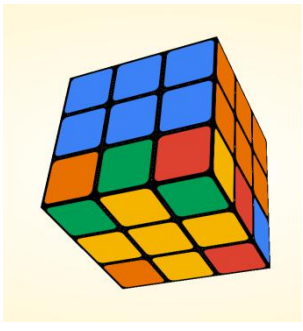
After creating the bottom cross we need to create yellow color in the bottom corner sides. Now we need to apply our 3rd algorithm. Both our second algorithm and third algorithm are part of the ‘OLL’ algorithms. Our third algorithm is-

r d R d r d d R (right hand algorithm)
or,
L D l D L D D l (left hand algorithm)

Here, also 3 situations can be possible. These are- 1 solved corner, no solved corner and 2 solved corners. In the case of 2 solved corners the solved parts can be positioned adjacently or, diagonally.

In the case of 1 solved corner we need to put the solved part in the left side in front of us and apply the right hand algorithm or, putting the solved part in the right side in front of us we need to apply the left hand algorithm. Both the cases will take us to either totally solved side or, one solved corner situated in another position. If we apply the 3rd algorithm in same way on that one solved corner will take us to totally solved side.

In the case where there is no solved corner there must be a side where the is no yellow pieces near to second layer like the picture below-



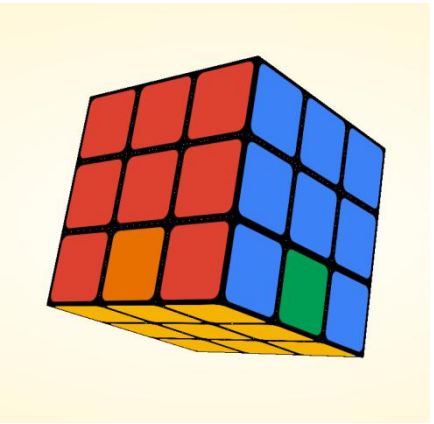
If we apply 3rd algorithm there that will bring us a one solved corner and for a one solved what we are needed to do we have just discussed.

In the case where there are 2 solved corners if solved parts are situated next to each other we need to put those solved corners in our rignt side and apply the right hand algorithm or, we need to put those solved part in the left side and apply the left hand algorithm. For diagonally positioned solved parts we can put one solved corner to our right side and can apply the right hand algorithm or, putting 1 solved corner to the left side we can use left hand algorithm. Both the works will take us to either 1 solved corner or, no solved corner. And now, we know what to do in those situations.

After solving bottom side's all the yellow colors we have only remaining the bottom sides near to second layer. Now we are needed to use our 4th algorithm and 4th and 5th algorithms are part of the 'PLL algorithm'. This algorithm is-

r B r F F R b r F F R R (right hand algorithm)
or,
L b L F F l B L F F L L (left hand algorithm)

We will use this algorithm to create headlights in all the sides. So, if there is at least 1 side that does not contain headlight we will match it with the second layer's center piece and will use this algorithm. If we have not found any headlight containing side we can apply the algorithm to any side which will create at least 1 side containing headlight. But if we have found that there is headlights and not all the sides contain headlights we are needed to put one headlight containing side in front of us and needed to apply either of the 2 algorithms above which will create headlights in all the sides. After getting that we are needed to match the headlights with the second layer's center pieces. Picture of a headlight is like this-



If we get headlights in all the sides automatically, we need not to use this algorithm.

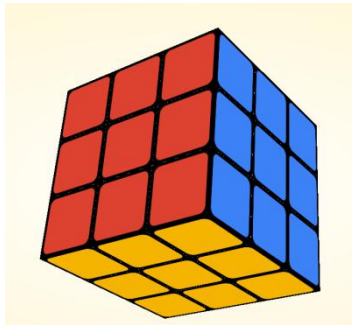
Now we are needed to use our 5th algorithm. This algorithm is-

F F (d/D) R L F F r l (d/D) F F

Here, **d/D** means depending on the center pieces it can be either **d** or, **D**. After doing **d** if the edge piece color matches with the center piece color then we need to use **d**. If **D** can do this then we are needed to use **D**. If neither **d** nor **D** can do this then we can chose anyone of the two. But in all the cases we need to remember the term and need to use it in the second **d/D** term.

To use this algorithm we are needed to see 3 situations. First of all, there can be a solved side which is called ‘a bar’ containing side or, the headlights are created in such a way where edge colors are matched with the opposite sides center pieces or, edge colors are matched with the adjacent sides center pieces.

If we find a bar we need to hold the rubik’s cube in such a way so that the bar containing side faces backward. In the case where 3rd layer’s edge pieces are matched with the adjacent sides center pieces, we need to hold the rubik’s cube in such a way so that one solved part faces backward and another solved part faces our left side. If we apply this algorithm now it will either solve the rubik’s cube or, will create another pattern like from edge pieces those are matching with the opposite sides center pieces to a bar sign situation. If we use this algorithm according to that situation that will solve our total rubik’s cube.



4.2 The Code

We have developed our code using python. Python is a well structured language so, though our code is long (near 7500 lines) but we did not get stretch. We have also added the part to run the stepper motor in the same code. That part contains about 200 lines of coding. Our main technique of running the stepper motor is we have saved our desired outputs in an array and then creating a while loop we have run our stepper motor running part. To add every step in the array we were needed to append the output in the array. So, if we do not consider those lines the length of our code becomes 40% lower but it is not possible. We have tested our code again and again for a lot of time and the code shows 100% correct output. We will attach the file containing the screenshots of our code with this report at the end.

To write the code we have followed our newly invented technique which is much easier to understand and implementing in the code. But here we also had to consider all the possible cases that we have mentioned before. So, in worst case, out of 64 possible cases only 4 cases will run.

Arm based computer like raspberry pi contains input output pins through which our stepper motor drivers(drv8825) are connected. With the drivers our stepper motors are connected. To use those pins the operating system has preinstalled libraries. Only we need to call the necessary libraries from our code.

```
import RPi.GPIO as GPIO
from time import sleep
```

To run the stepper motors we were needed to define the step pins and direction pins in the code. As we have used micro stepping to run the motors smoothly we were also needed to define those pins in the code.

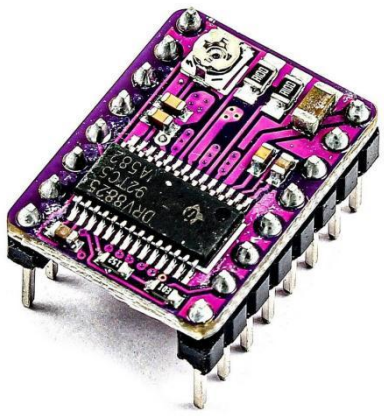
Sample screenshot of our code for running raspberry pi is as below:

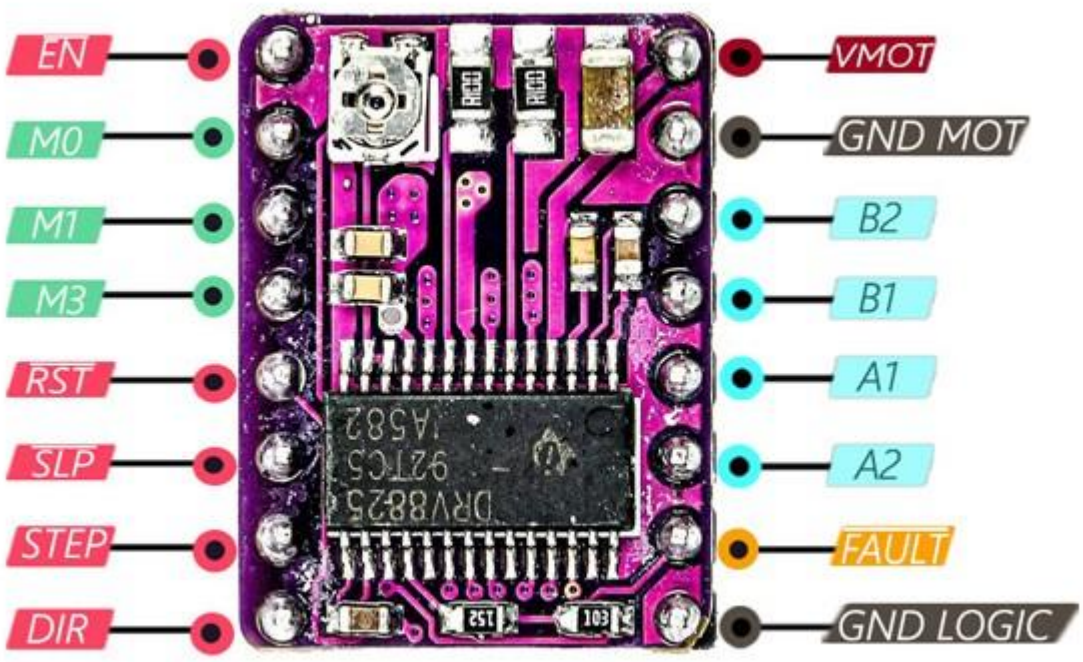
```
# for Raspberry pi

CW = 0     # Clockwise Rotation
CCW = 1    # Counterclockwise Rotation
SPR = 50   # Steps per Revolution (360 / 1.8)

# Defining pin numbers

STEP_L = 23
DIR_L = 24

STEP_R = 12
DIR_R = 16

STEP_B = 20
DIR_B = 21

STEP_F = 10
DIR_F = 9

STEP_D = 19
DIR_D = 26

# Defining MODE (Microstep Resolution GPIO pins)

MODE_L_MOTOR = (14, 15, 18)
MODE_R_MOTOR = (25, 8, 7)
MODE_B_MOTOR = (2, 3, 4)
MODE_F_MOTOR = (17, 27, 22)
MODE_D_MOTOR = (5, 6, 13)

sleep(5)
```

```
i=0
for i in range (len(result)) :
    if result[i]=='L' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_L, GPIO.OUT)
        GPIO.setup(STEP_L, GPIO.OUT)
        GPIO.output(DIR_L, CW)
        GPIO.setup(MODE_L_MOTOR, GPIO.OUT)
        GPIO.output(MODE_L_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_L, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_L, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='l' :
```

4.3 Machine setup

Before describing our setup we need to show some basic knowledge on how to use drv8825 stepper motor driver. Below is a picture of a drv8825 driver-





Here, EN (enable pin) is by default set to on. If want to stop working of that pin we need to connect that pin to ground which will stop the whole working process of the driver.

M0, M1, M2 are the pins for the micro stepping mode.

Enabling RST (reset pin) will reset all the previous setup. We should connect the pin with raspberry pi's 3.5 volt pin.

SLP (sleep pin) pauses all the works on the driver for the time we will set in our program. To use this pin we need to connect this pin with the raspberry pi's 3.5 volt pin.

STEP pin is used to run the stepper motor's single step (in our stepper motor this step is 1.8 degree). as we will work on this STEP pin in the program we need to connect it with the raspberry pi's i/o pins.

DIR (direction pin) give direction that the motor will rotate clockwise or, anticlockwise. As we need to program it we need to connect this pin with the raspberry pi's i/o pins.

VMOT is the positive connection part of our external voltage source. In our case we have used 24 volt and 1.5 amp external voltage source.

GND MOT is the negative connection part of our external voltage source. We should make a rail with the raspberry pi's ground.

A1, A2 and B1, B2 are the pins for two motor coils pairs. We should connect our stepper motor's 4 wires with this pins after finding out the wires from the same coil correctly.

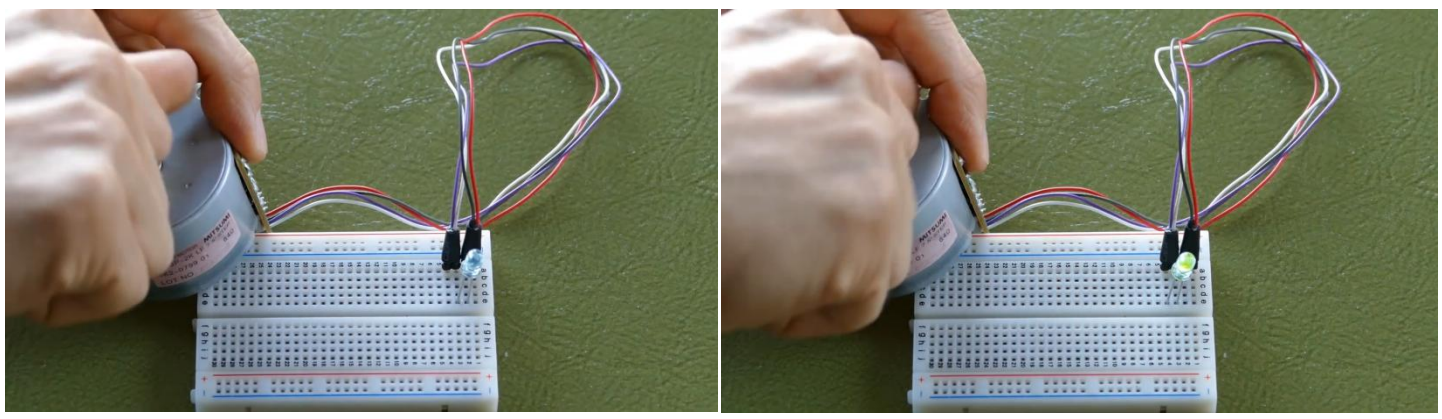
FAULT pin is not used in general cases.

GND LOGIC pin should be connected with the raspberry pi's ground pin.

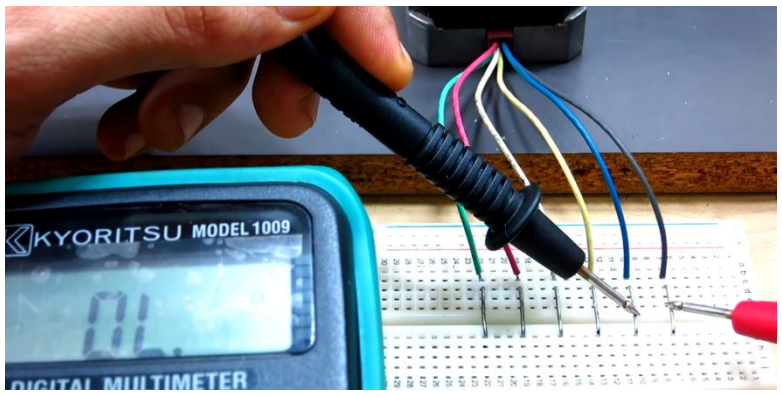
We can also notice that there is a screw in the top left side of the driver. By rotating clockwise it lowers down the current providing in the driver and rotating anticlockwise it increases the current. To continuously supply current drv8825 driver can provide maximum 1.5 ampere current and over 2.2 ampere current can destroy the driver. This driver supports at most 45 voltage over which will destroy our driver. So, we need to be careful on providing correct amount of current.

4.3.1 How to identify wires from the same coil of the stepper motor

We can easily find it with a small led light. After connecting it with the 2 wires when we will rotate the motor with hand if the led turns on it means those coils are from the same coil.



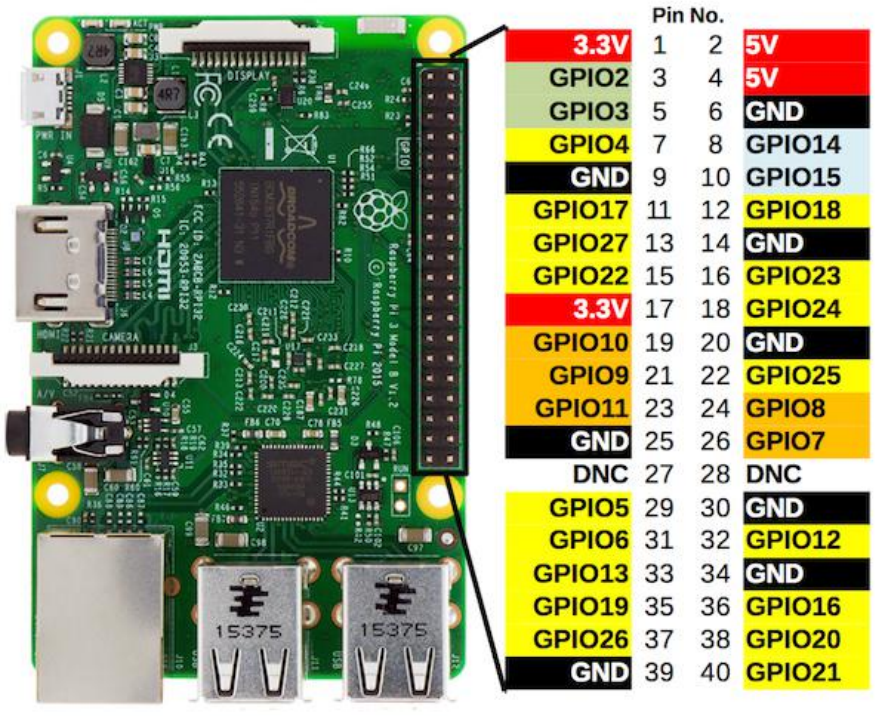
This process works well with the 4 wire bipolar stepper motor but for the 6 wire bipolar stepper motor we also need to identify the 2 neutral wires. To do this we need to use a multi meter.



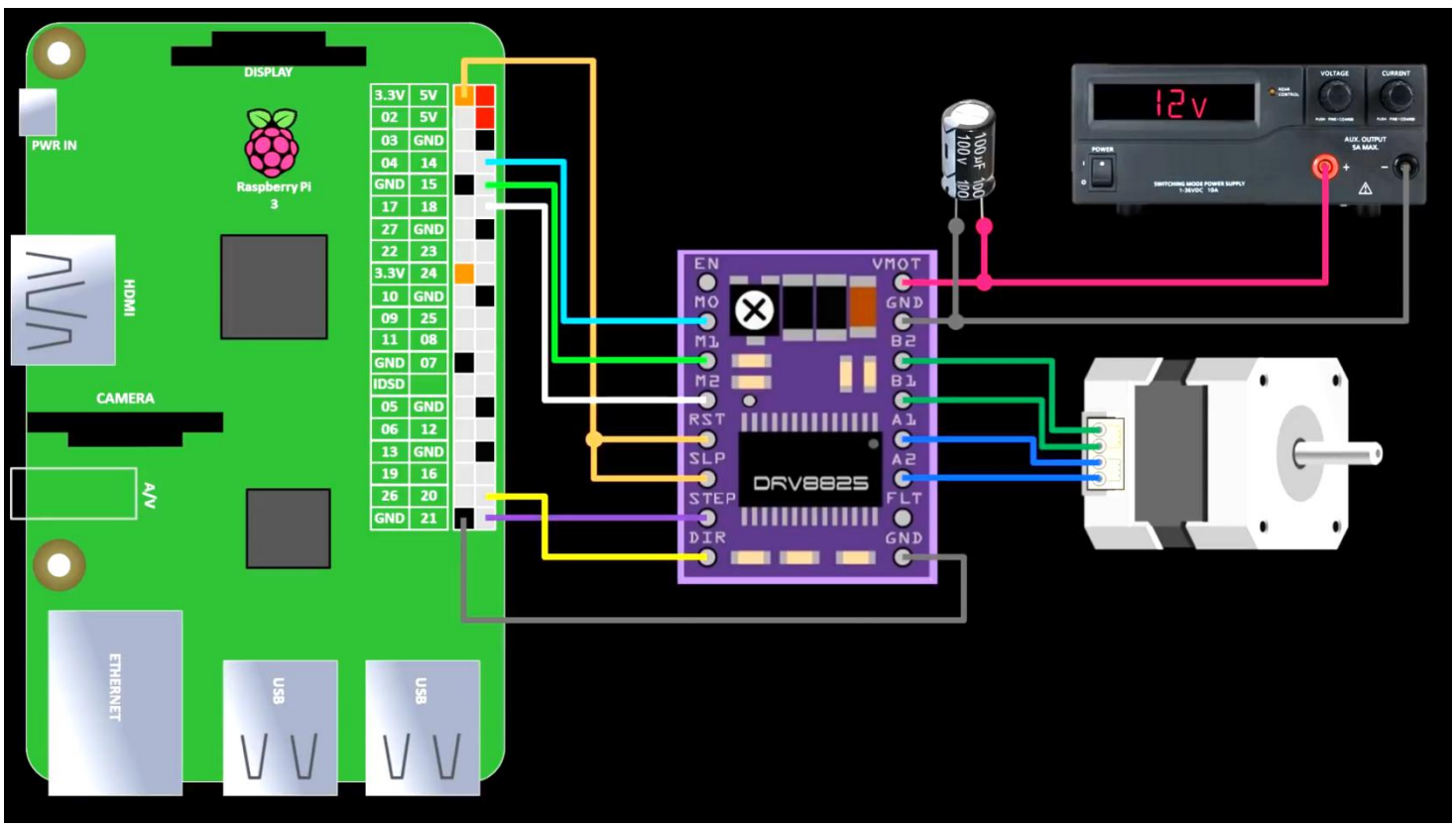
Wires from the same coils will show reading on the multi meter. Along the three wires from the same coil we will get 3 values on the multi meter. Middle valued wire is our one neutral wire. Among the other 3 wires we will get 3 values on the multi meter. Here, also the middle valued wire is the neutral wire. In most of the cases white and yellow wires are the neutral wires.

4.3.2 Raspberry pi setup

To setup raspberry pi we need to describe the pins on the raspberry pi. There are 40 i/o pins in the raspberry pi. But not all the pins can be used as i/o pin because there are some ground pins, one 3.3v pin, two 5v pins and some other pins. More importantly we need to know the GPIO pin numbers to write the code but it is not labeled on the raspberry pi.

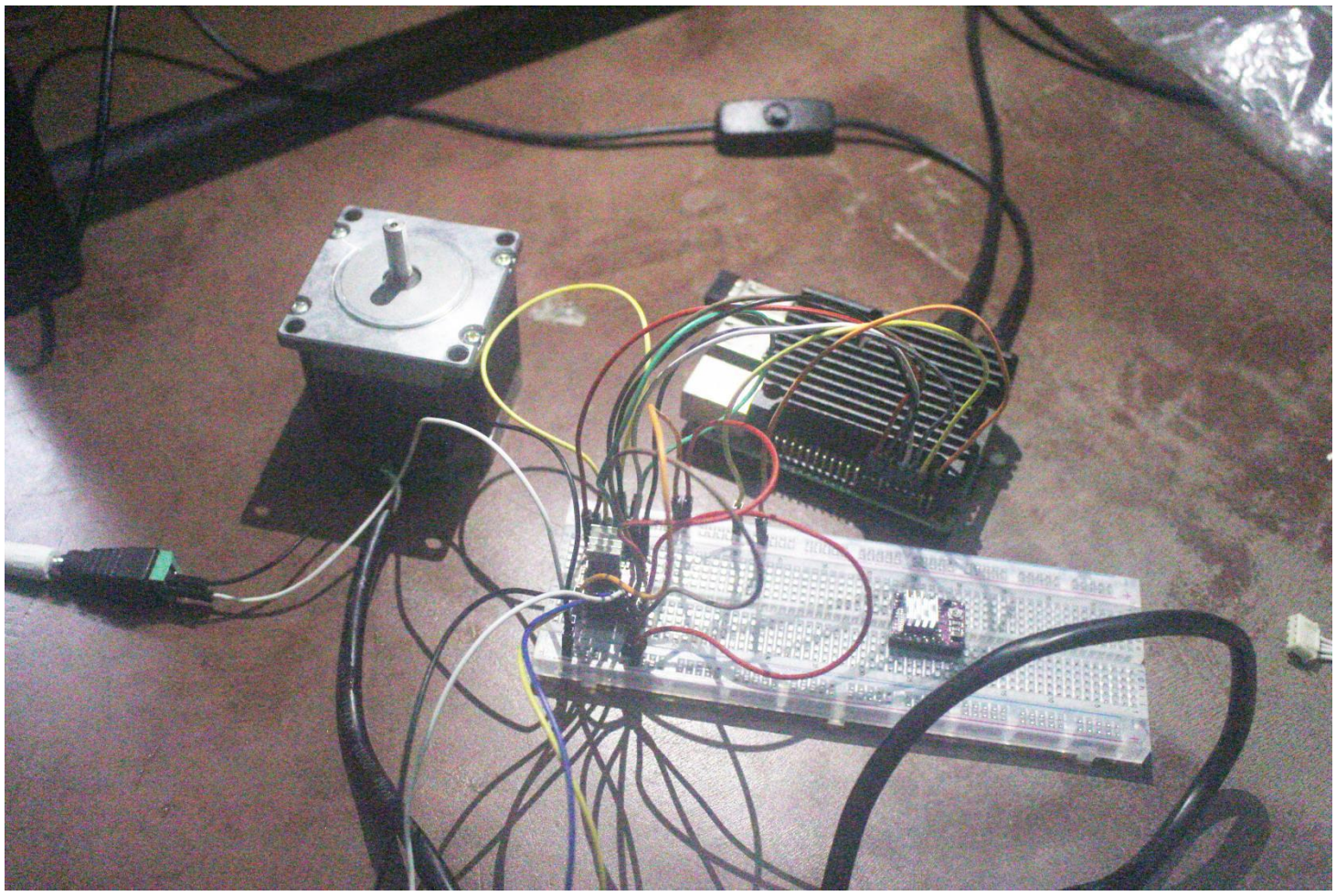


Our total setup for running one stepper motor is like the diagram below-



Chapter Five
Project Setup Pictures and Description

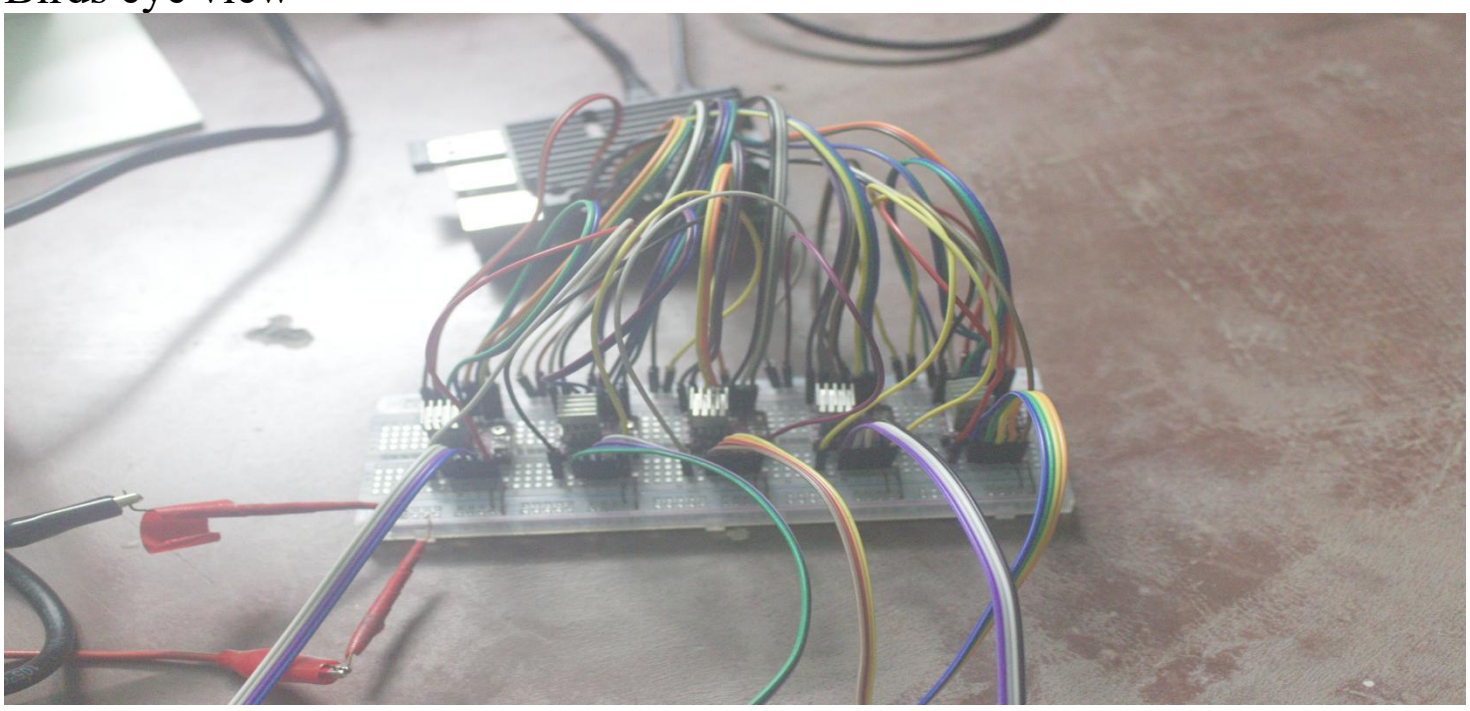
In our practical setup the picture for running only one stepper motor is like below-



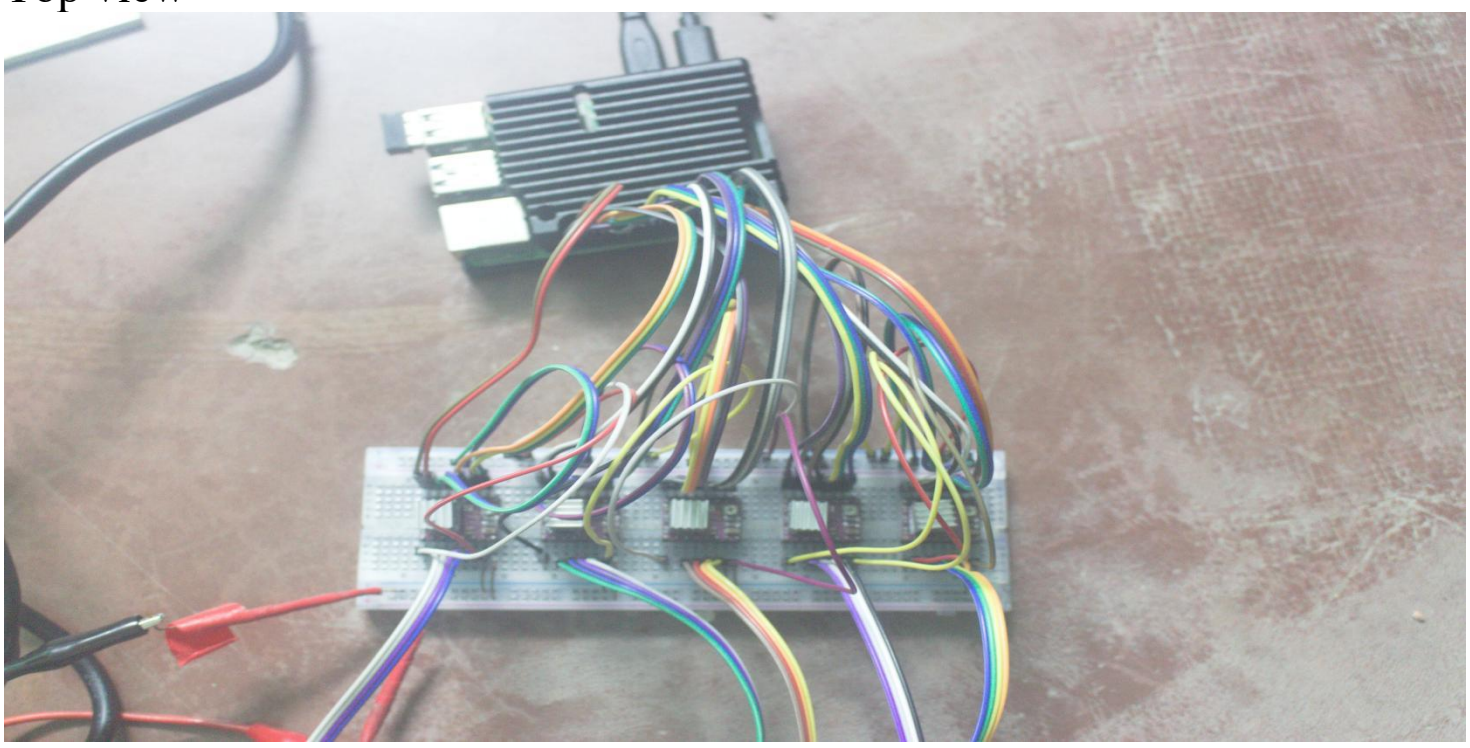
Here, we can see running only one stepper motor is also required to setup all the necessary equipment like we need to setup our raspberry pi, we need to write and run a code, we need to provide external power, connecting the wires correctly on a breadboard in a word everything. So, it is clear that by learning how to run only one stepper motor we can run all the 5 stepper motors at a time.

After testing one stepper motor, when we were able to run that properly we have setup the system for our large project. The setup for our large project is like the picture below-

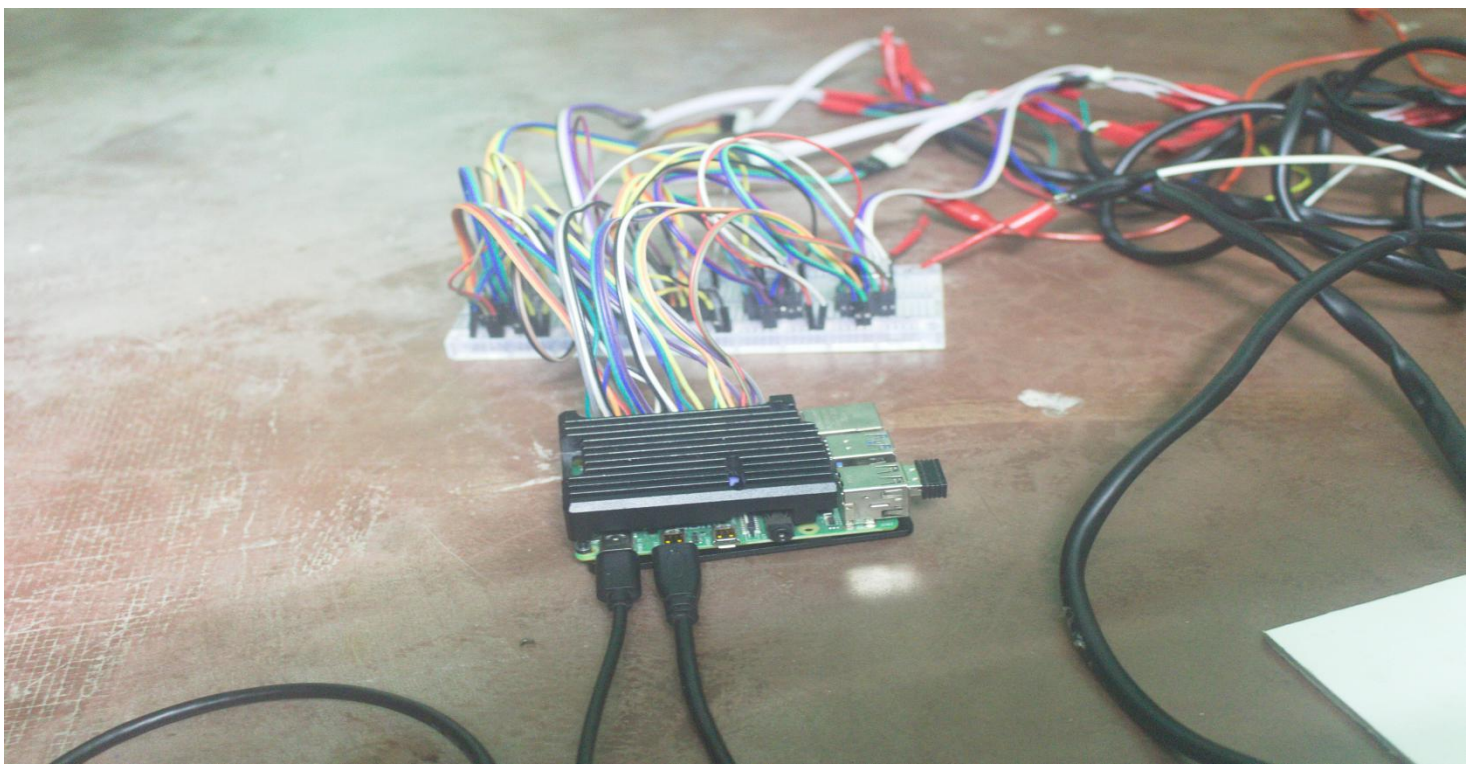
Birds eye view-



Top view-

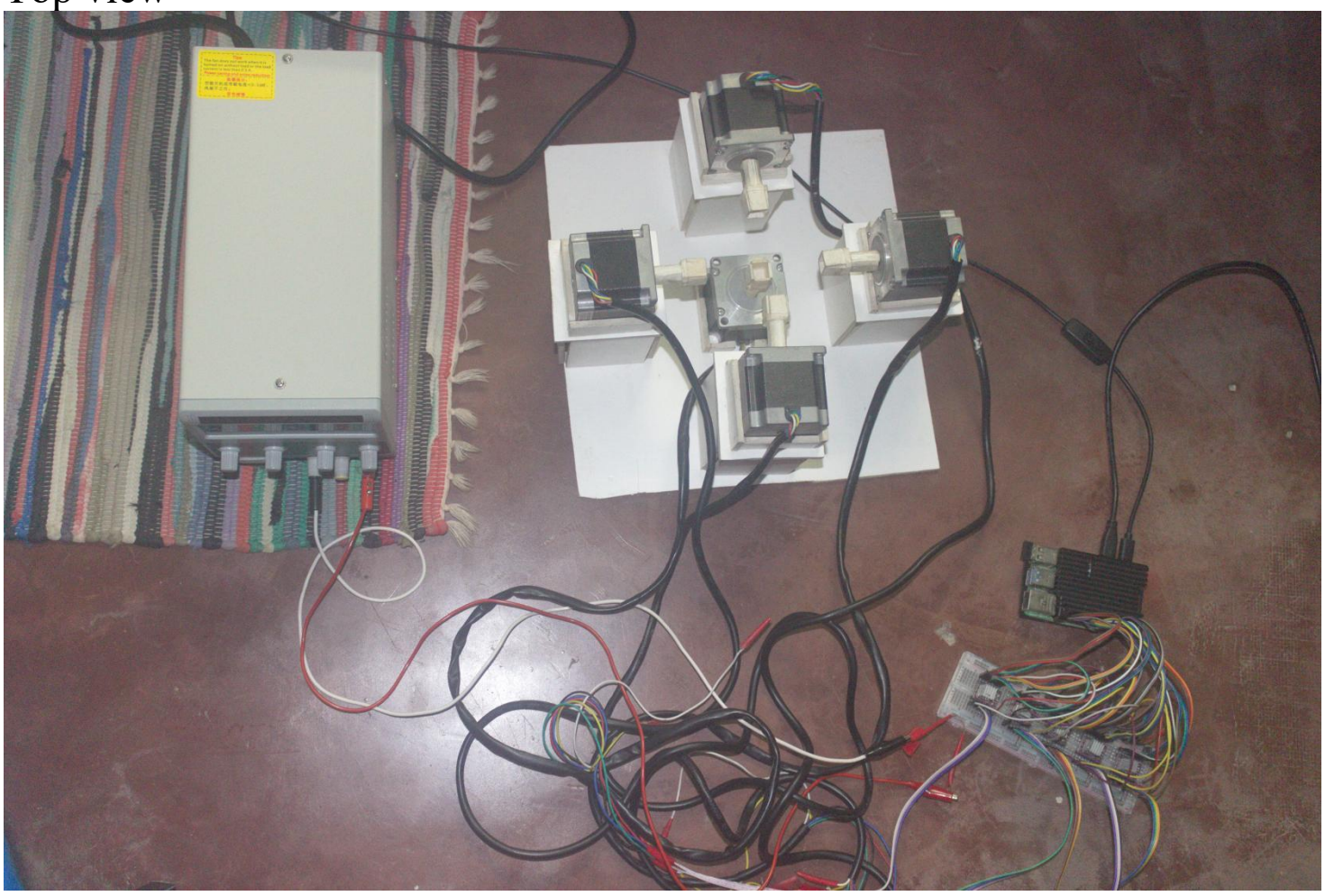


Arial view-

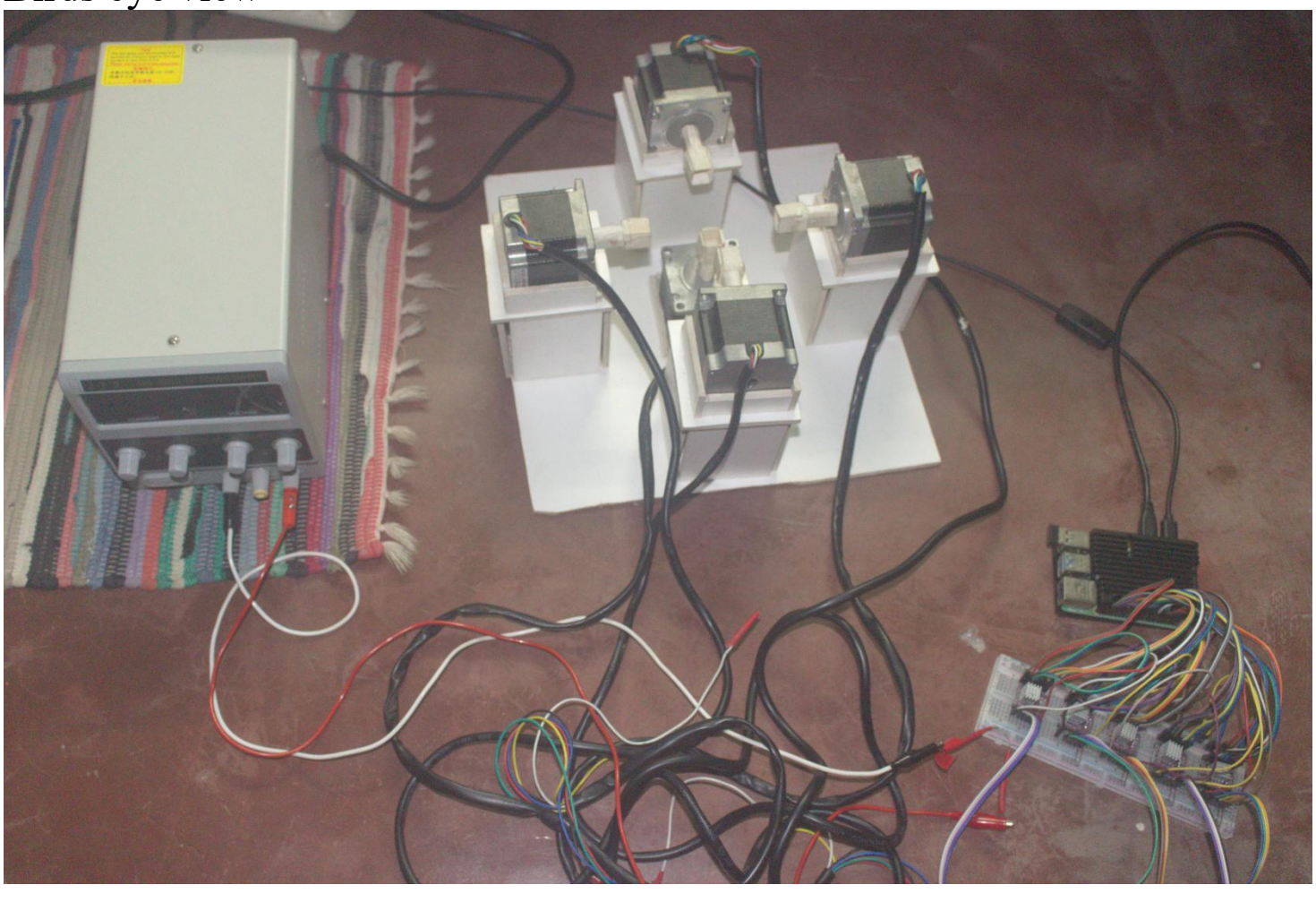


After successfully running all the 5 stepper motors at a time we were needed to make a frame to put our stepper motors properly to rotate our rubik's cube properly. The frame picture is like below-

Top view-

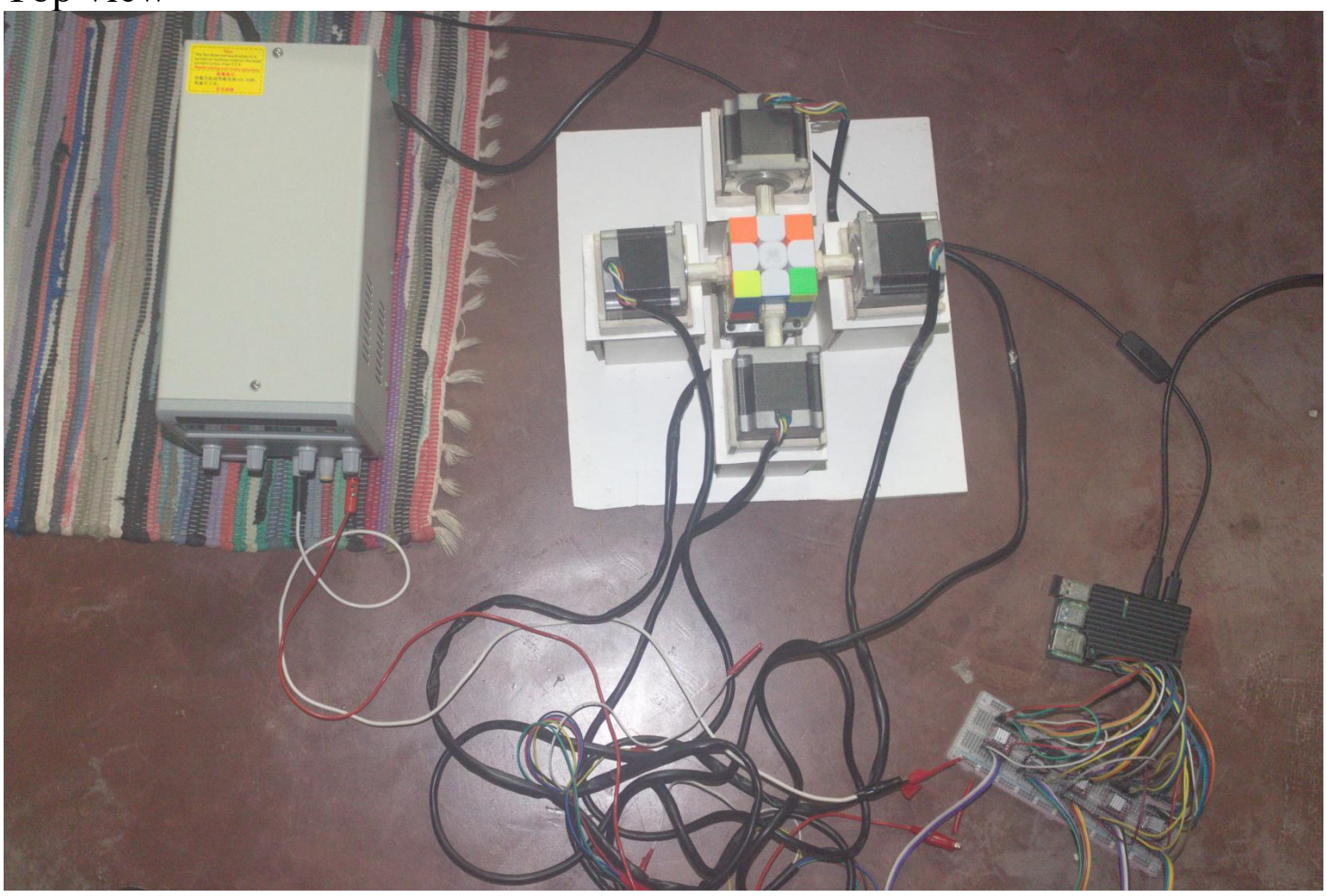


Birds eye view-

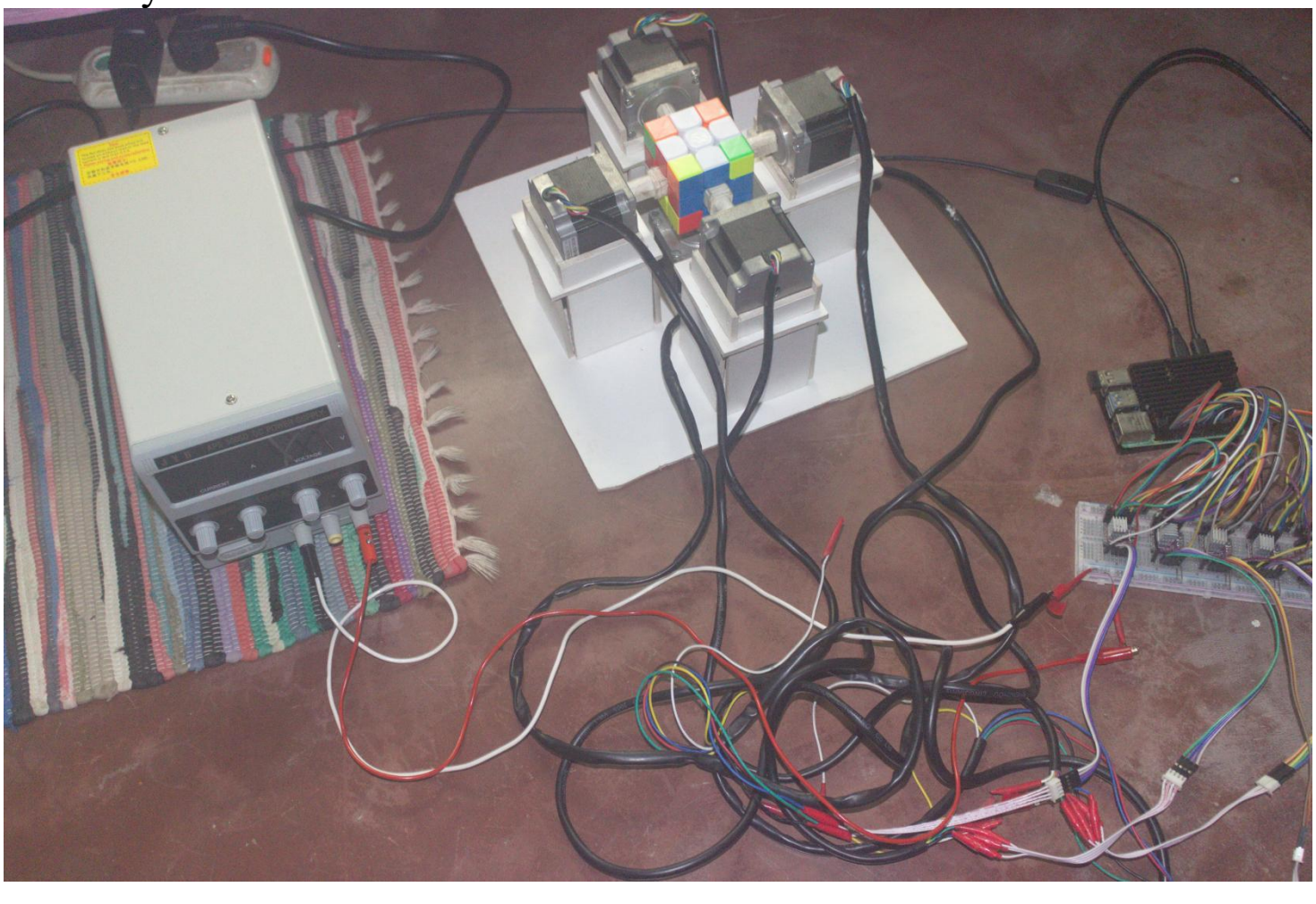


After making the frame we have tested whether all the stepper motors are running properly or, not on that frame. After successfully testing we have tested our final code without any rubik's cube. After seeing positive result we have tested out our final project that is giving a rubik's cube inside the frame. At that moment the picture is like below-

Top view-

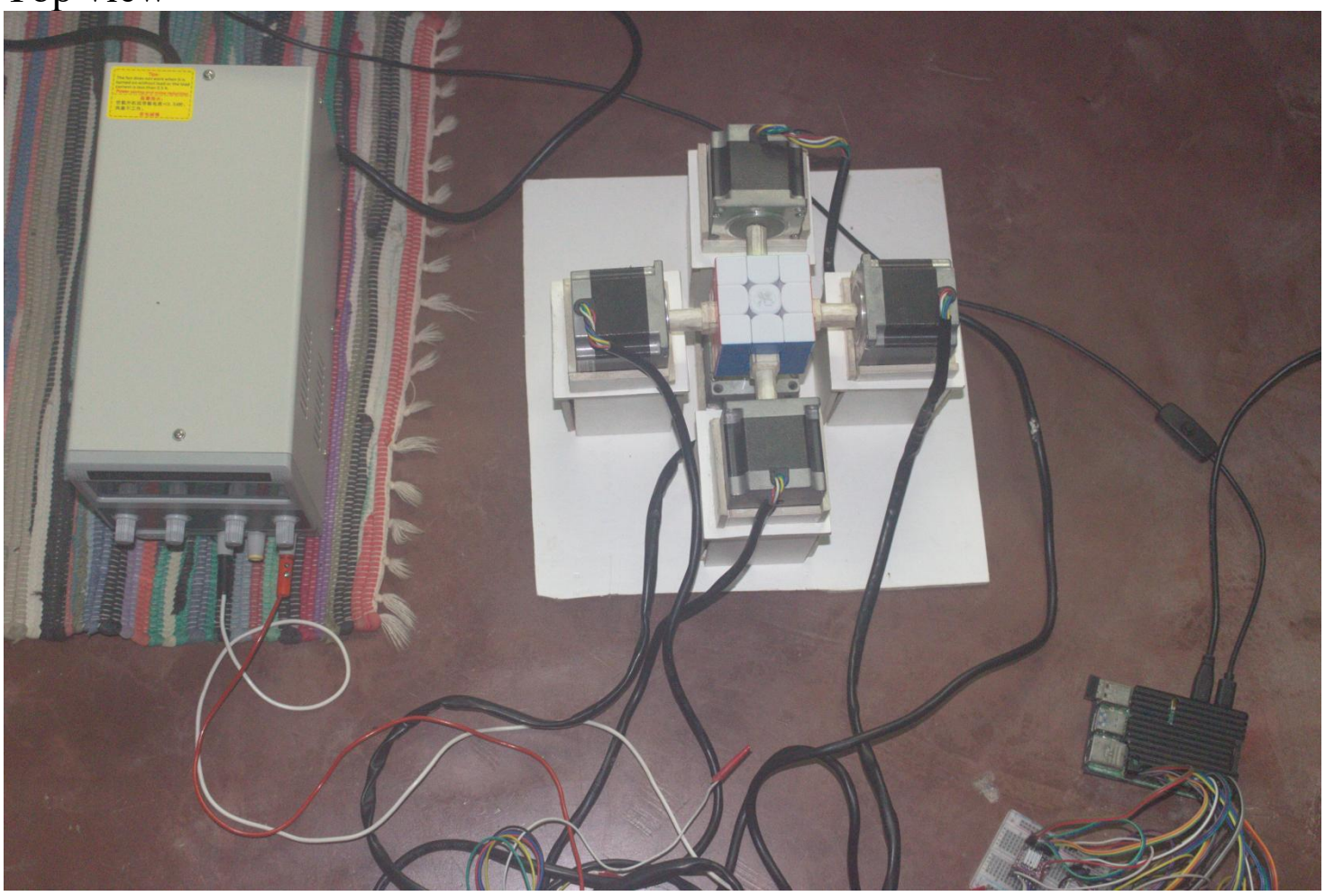


Bird's eye view-

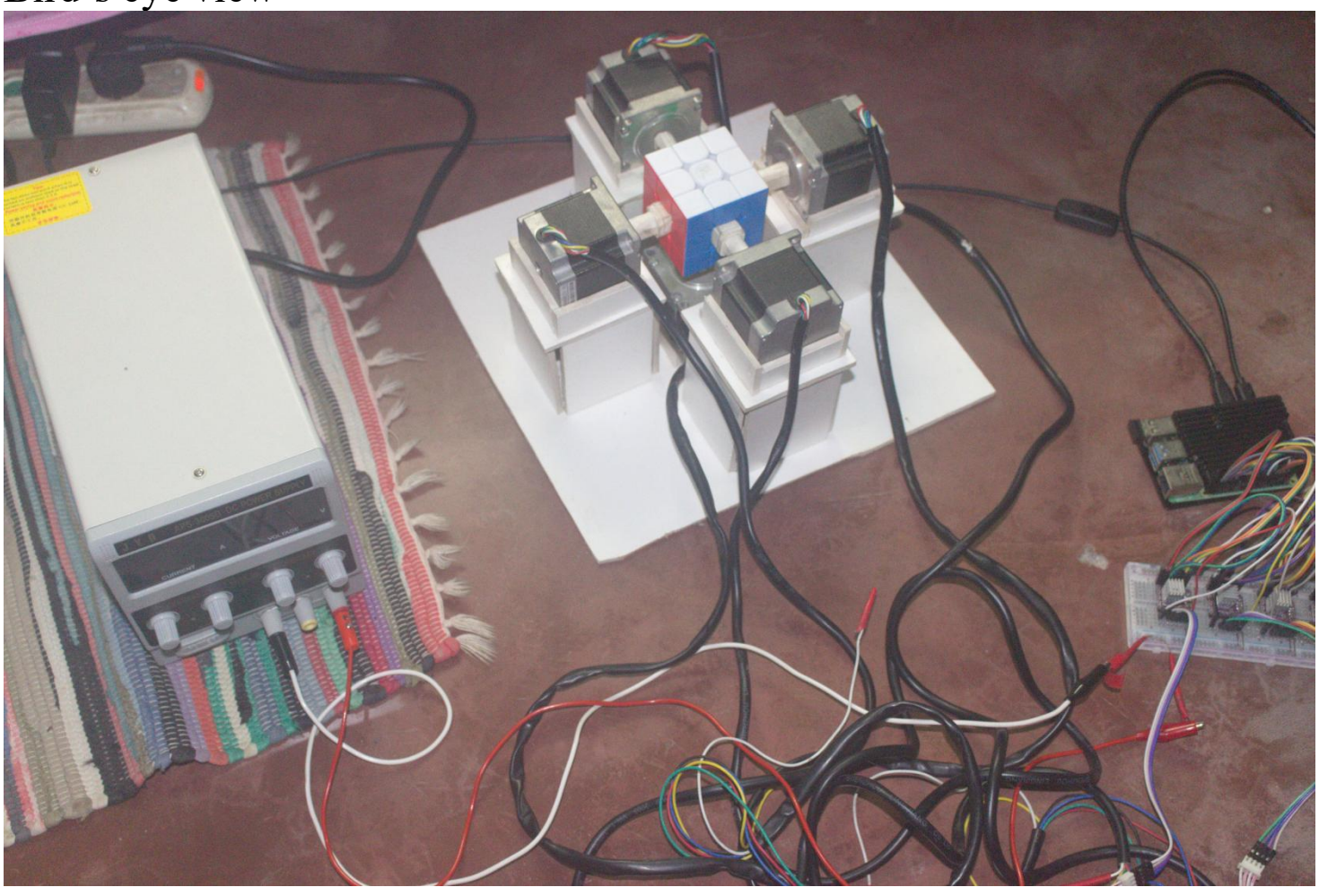


After running the code our rubik's cube solver machine had started running. After finishing the solving the picture is like below-

Top view-



Bird's eye view-



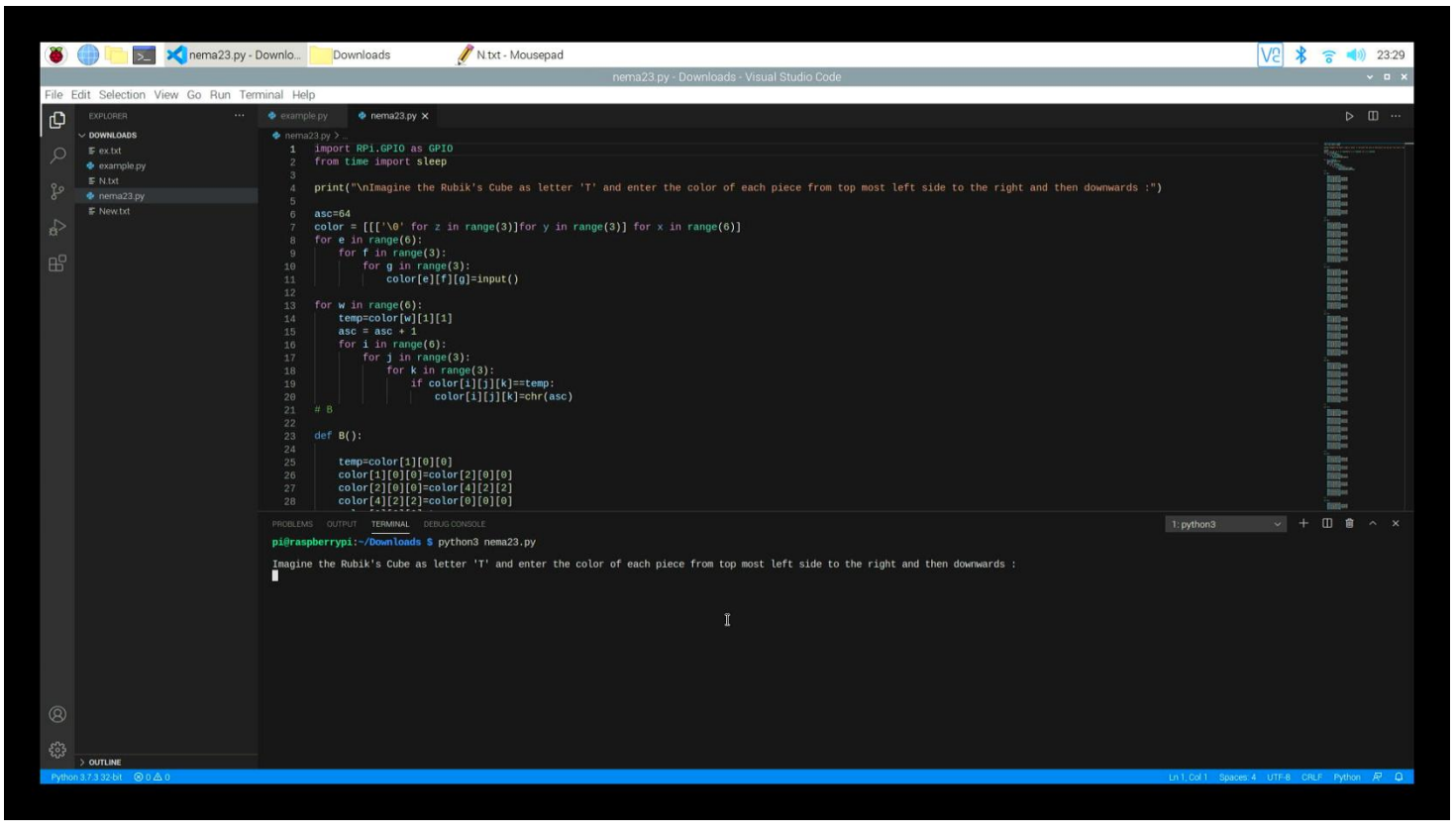
5.1 Proof of correct output of the code

To show our code is fully functioning we can see the outputs of our code and solve a rubik's cube manually. As our code is one of the main topic in this project we want to show the proof that it is running nicely. Let, we have a scrambled cube and it's current situation is-

b y o w g r r o o g w o b w w g y r b y r r b g y w b y r g b r o g o w w g r g y r w b y o y b g o o w b y

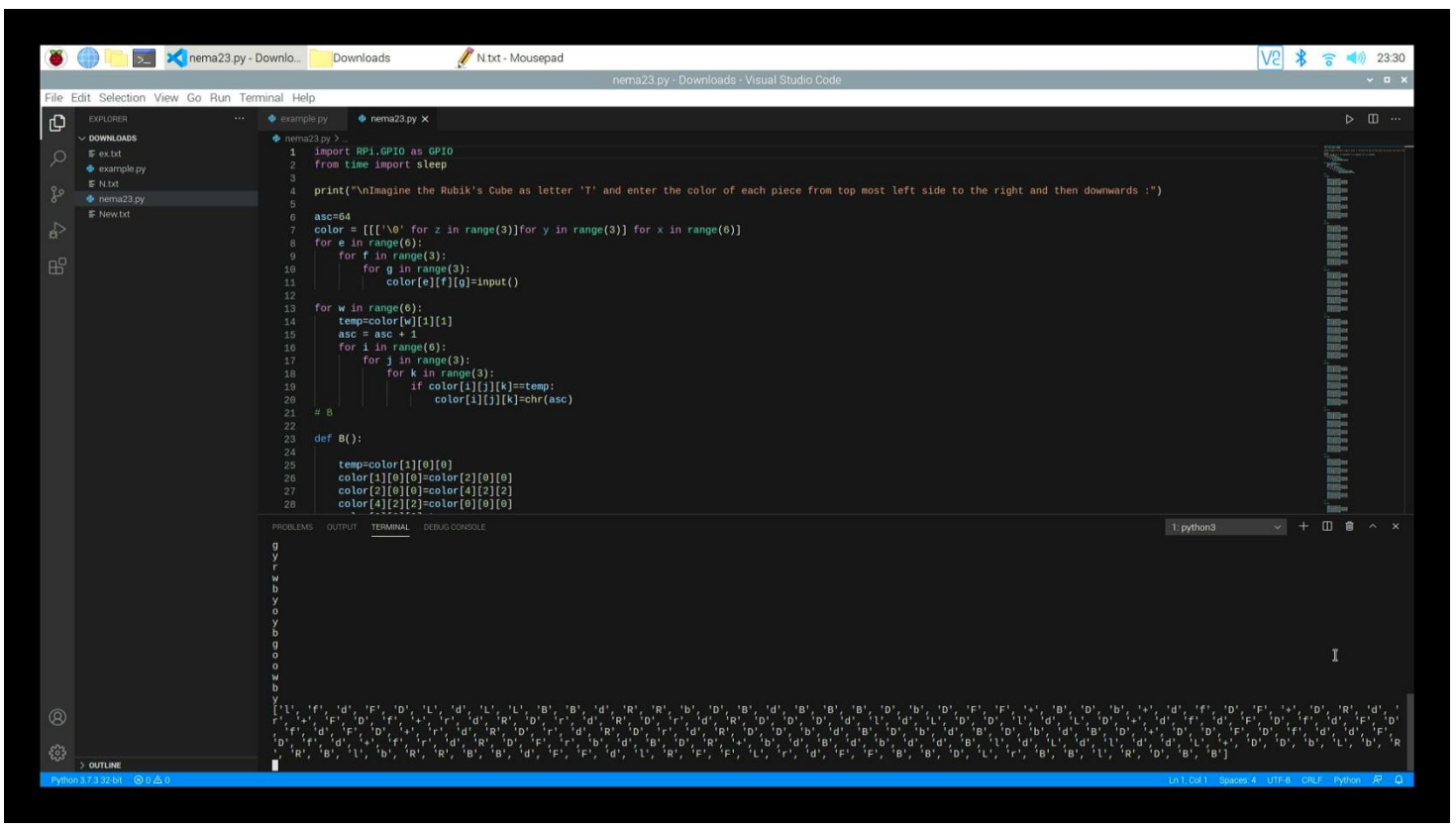
Here, w = white
r = red
b = blue
g = green
y = yellow
o = orange

This positioning starts from the left side, then top side, then right side, then front side, then down side, then back side so that it creates a capital letter 'T' and positioning starts in every side from the top most left side to the right and then go downward. After running the code the output shows like this-



The output is showing "Imagine the rubik's cube as letter 'T' and enter color of each piece from top most left side to the right and then downwards :"

Here, one thing we want to mention is that generally, all the python IDE run on python2 by default. To run any code with python3 we need to set command line in the terminal. That's why we have written in the terminal 'python3 nema23.py'. 'nema23.py' is our file name. After giving inputs which we have mentioned above the code is showing the output like this-

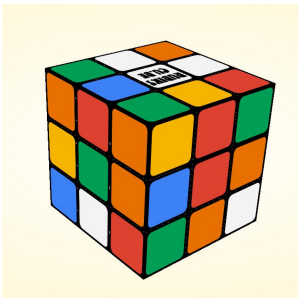


The output it showing is-

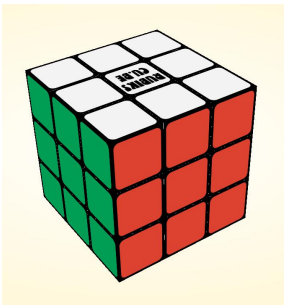
l f d F D L d L L B B d r r b D B d B B B D b D F F + B D b + d f D F + D R d r + F D f + r d R D r d R D r d R D r d R D D D d l d L D D l d L D + d f d F D f d F D f d F D + r d R D r d R D r d R D r d R D D b d B D b d B D b d B D + D D F D f d d F D f d + f r d R D F r b d B D R + b d B d b d d B l d L d l d d L + D D b L b R R B l b R R B B d F F d l R F F L r d F F B B D L r B B l R D B B

Here, L = left side clockwise rotation.
l = left side anticlockwise rotation.
R = right side clockwise rotation.
r = right side anticlockwise rotation.
B = back side clockwise rotation.
b = back side anticlockwise rotation.
F = front side clockwise rotation.
f = front side anticlockwise rotation.
D = down side clockwise rotation.
d = down side anticlockwise rotation.
+ = it is used for the users to follow the output steps easily.

So, our initial condition is-

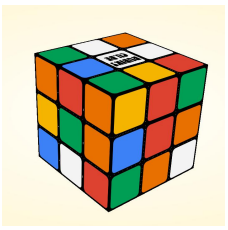


And our goal is-

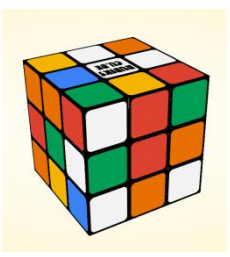


Lets do it by following the output-

Initial condition-



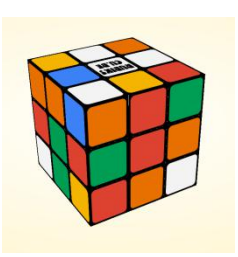
l



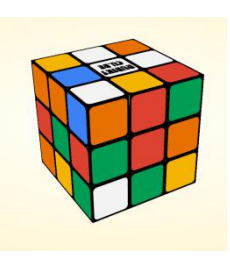
f

d

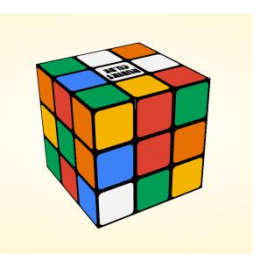
F



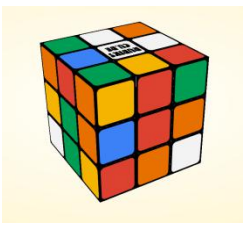
D



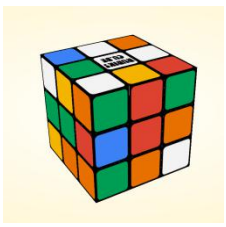
L



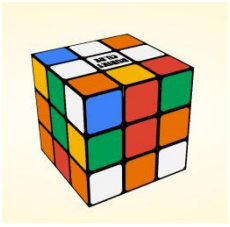
d



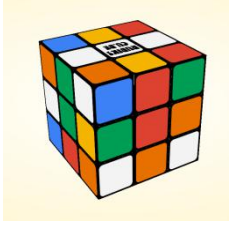
L



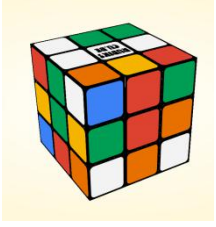
L



B



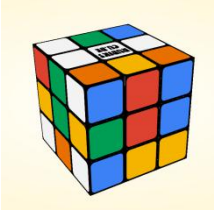
B



d

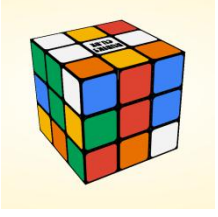
R

R

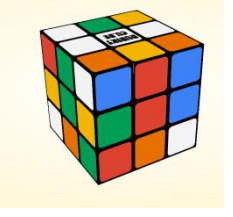


b

D

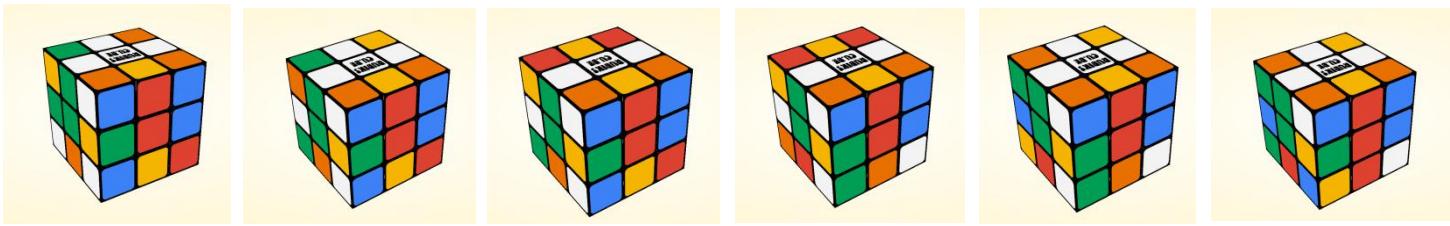


B

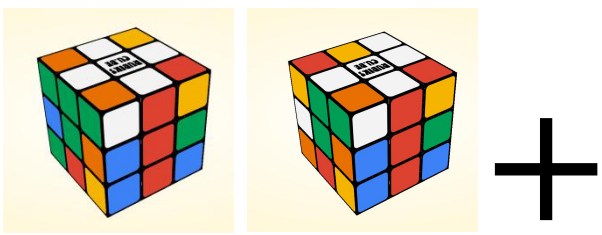


d

B B B D b D

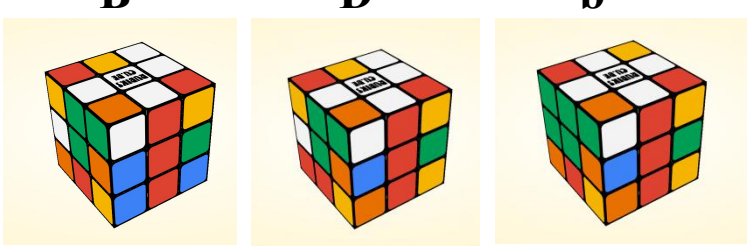


F F



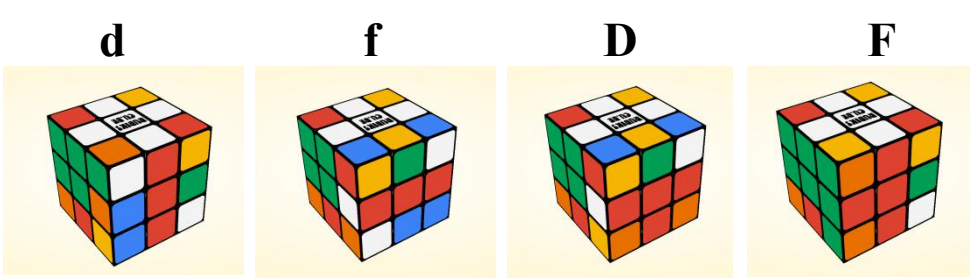
+

B D b



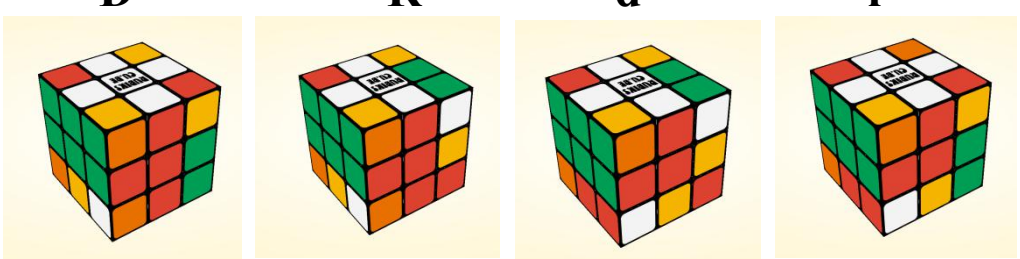
+

d f D F



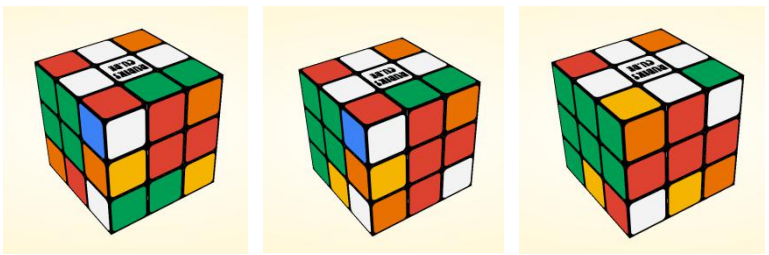
+

D R d r



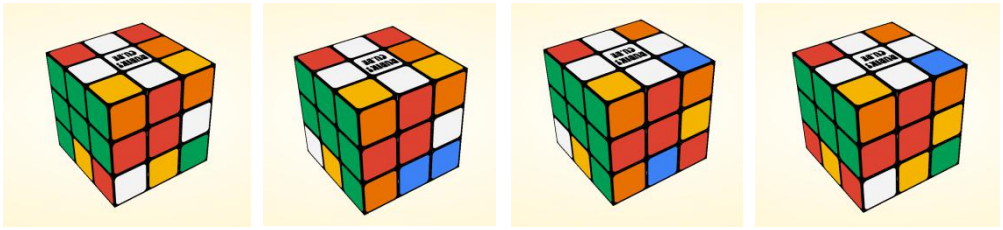
+

F D f

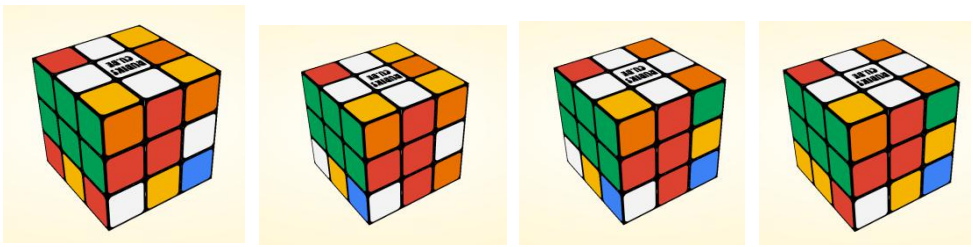


+

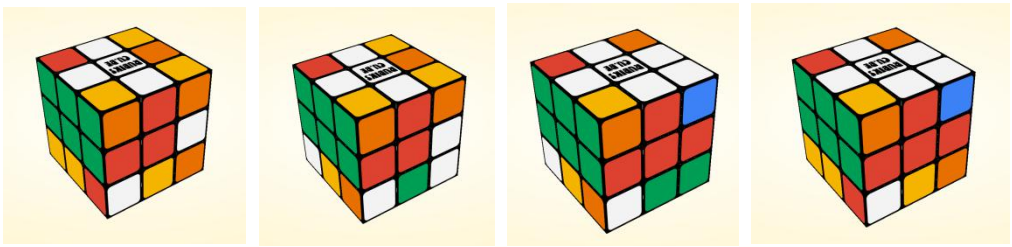
r **d** **R** **D**



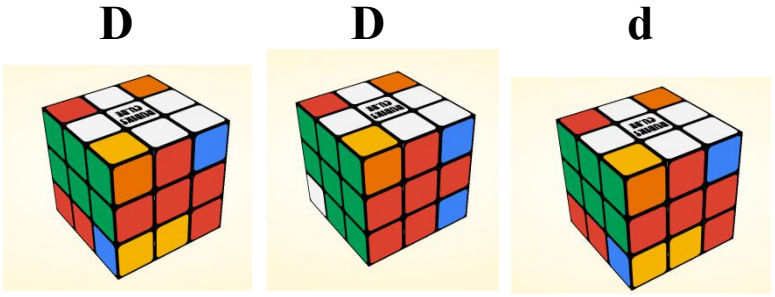
r **d** **R** **D**



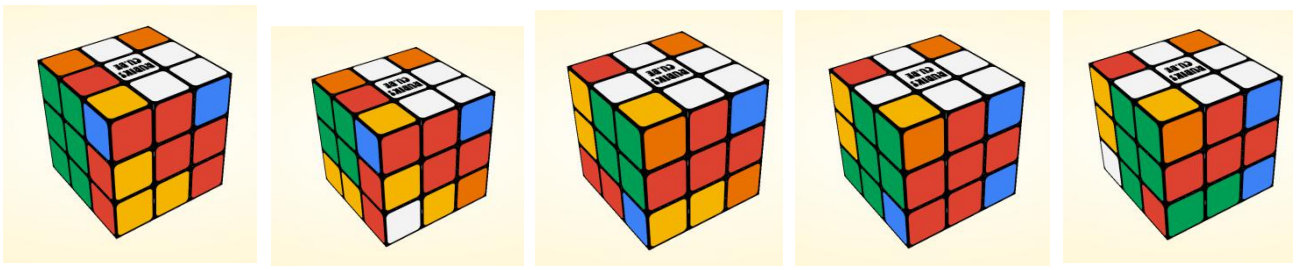
r **d** **R** **D**



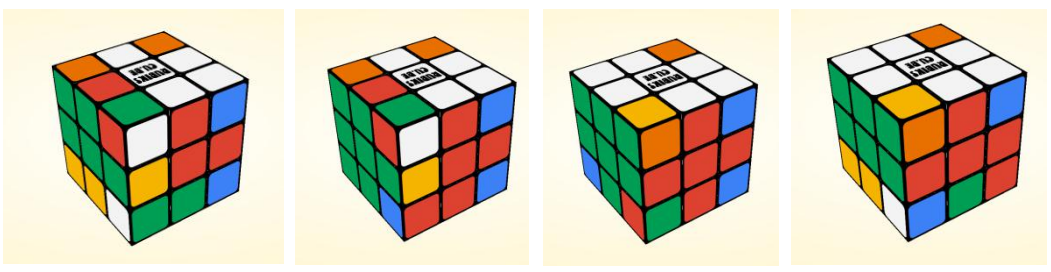
D **D** **d**



l **d** **L** **D** **D**

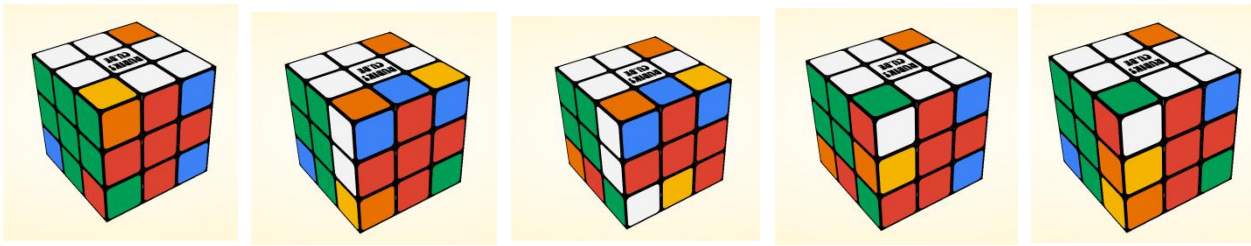


l **d** **L** **d**

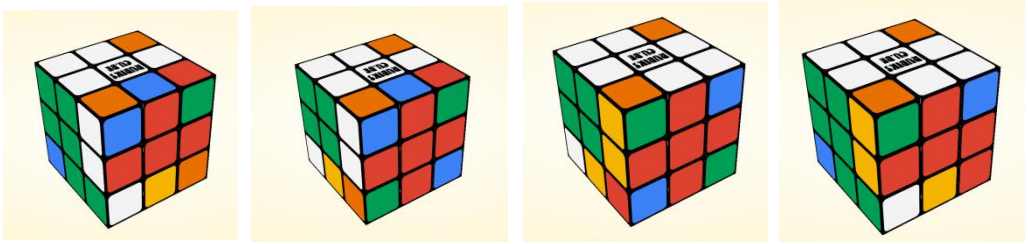


+

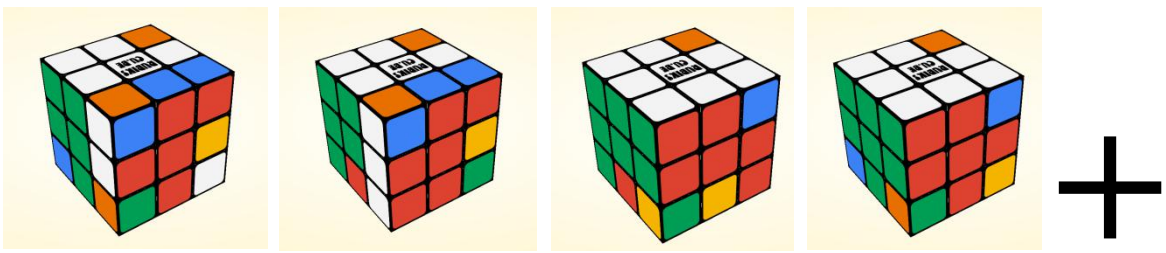
d f d F D



f d F D

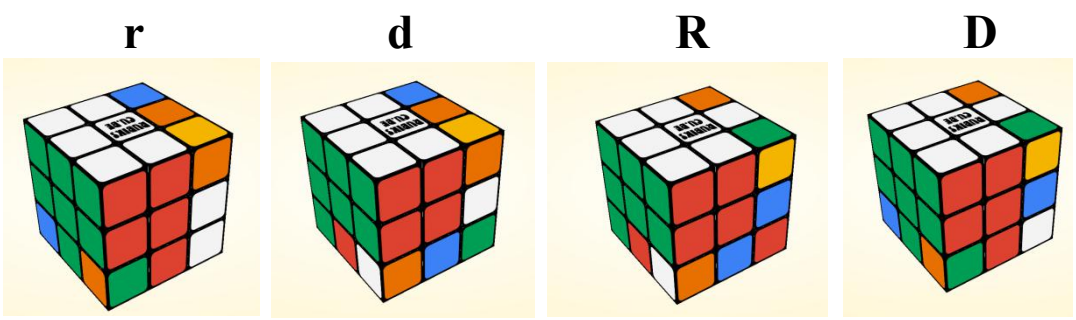


f d F D

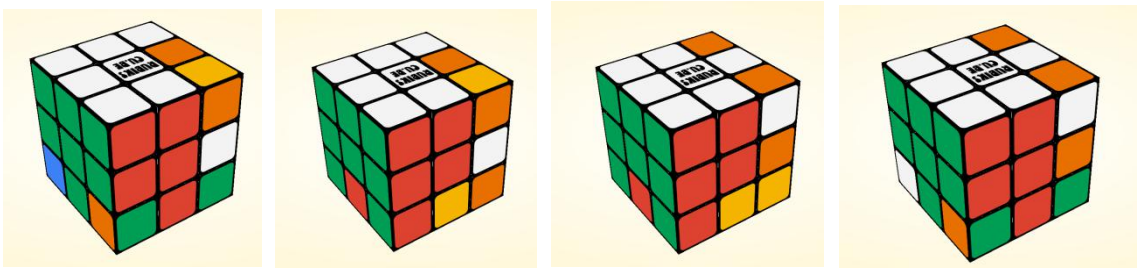


+

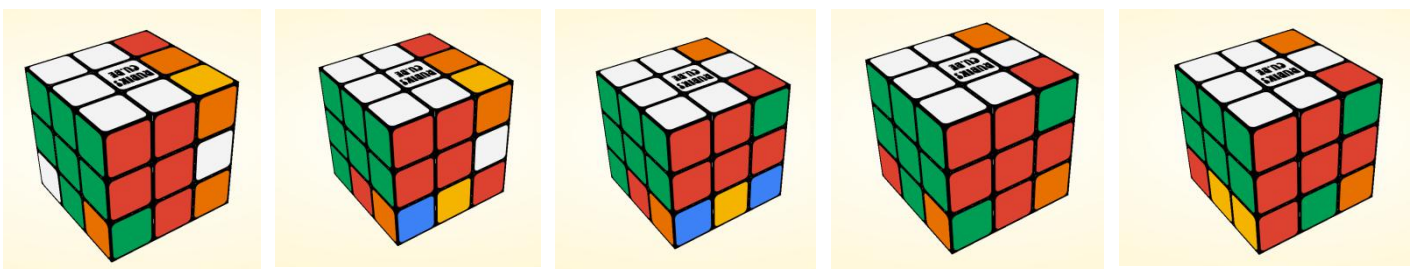
r d R D



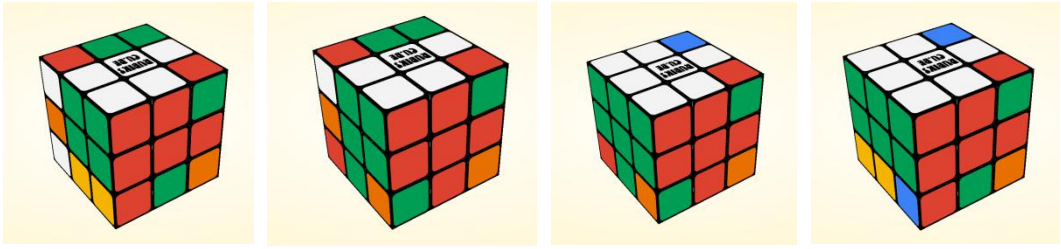
r d R D



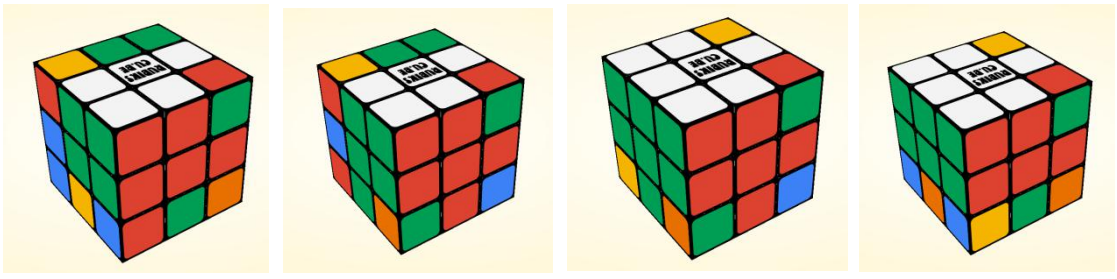
r d R D D



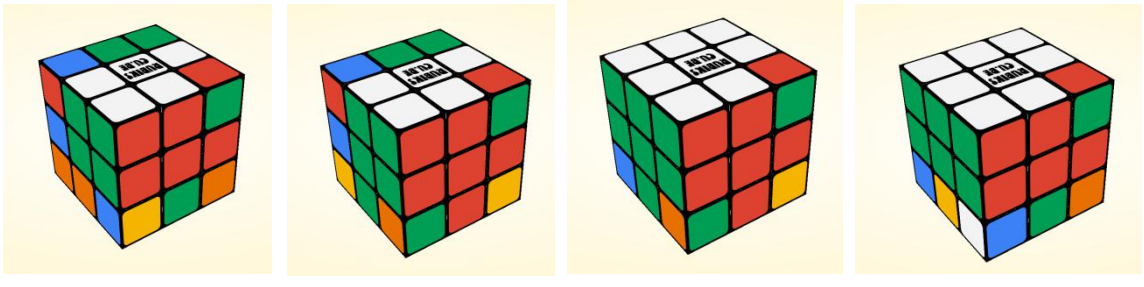
b d B D



b d B D

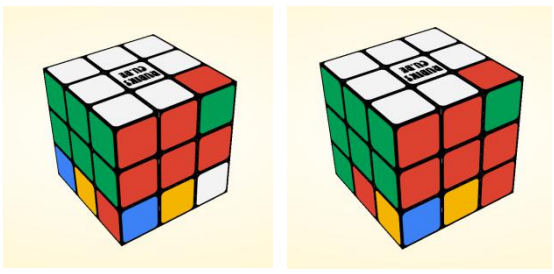


b d B D

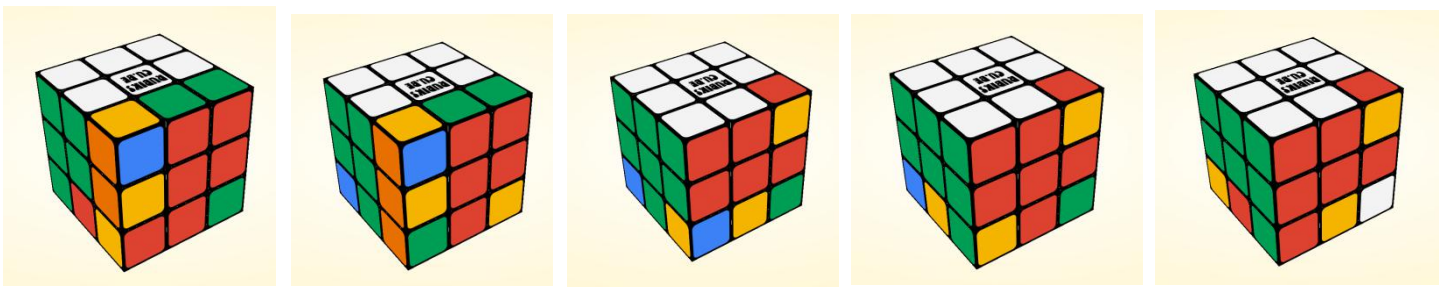


+

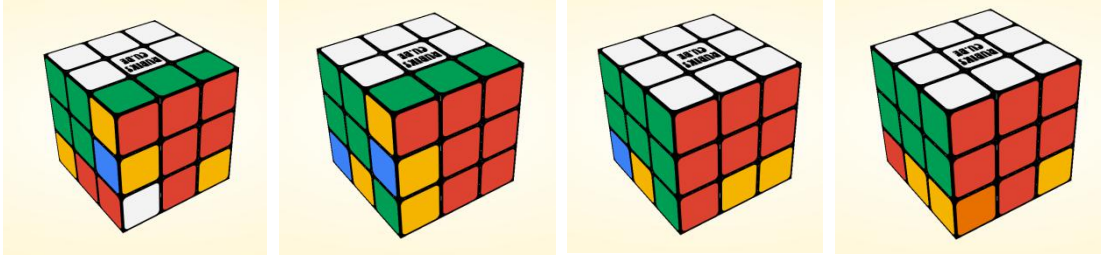
D D



F D f d d

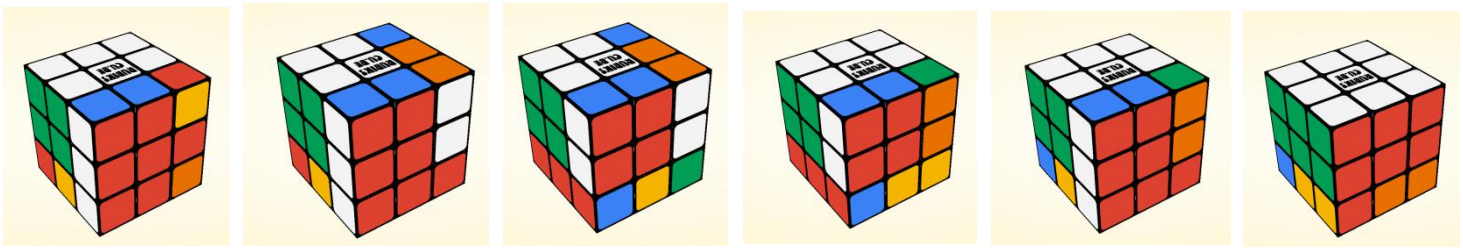


F D f d

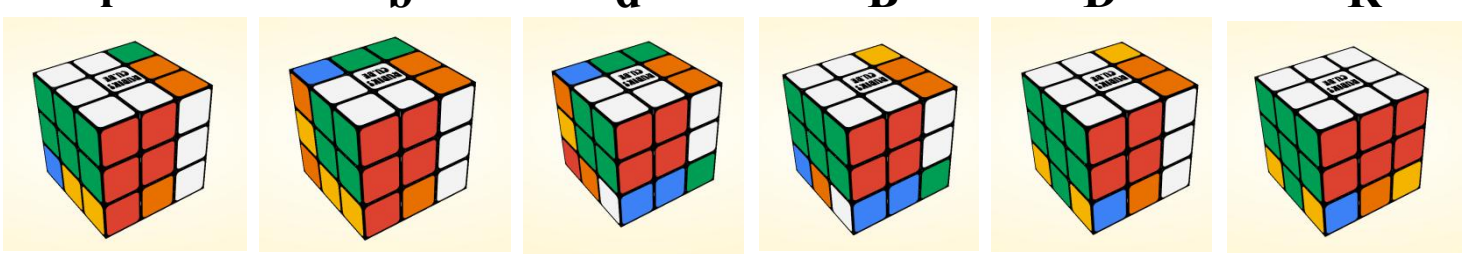


+

f r d r d f

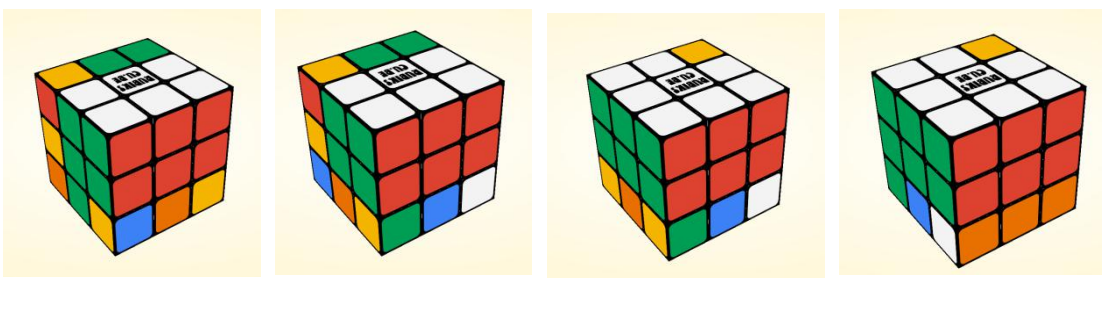


r b d B D R

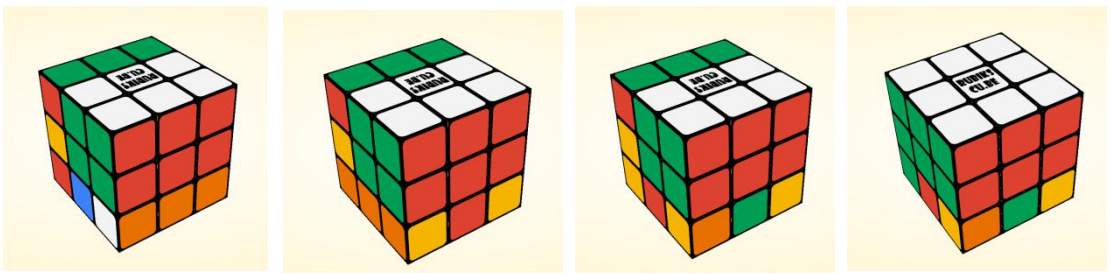


+

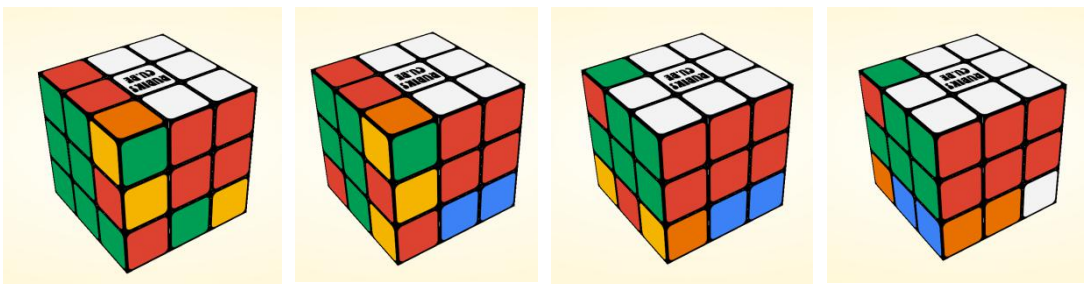
b d B d



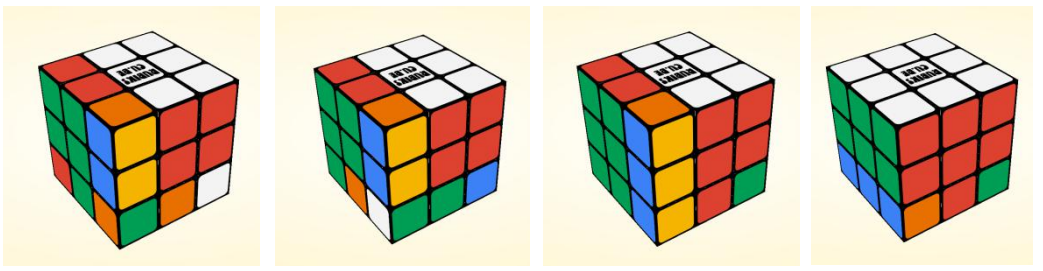
b d d B



l d L d

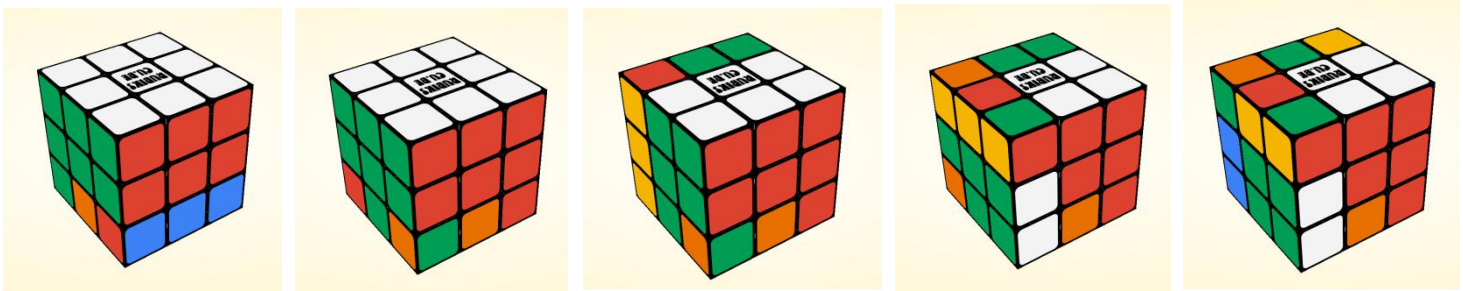


l d d L

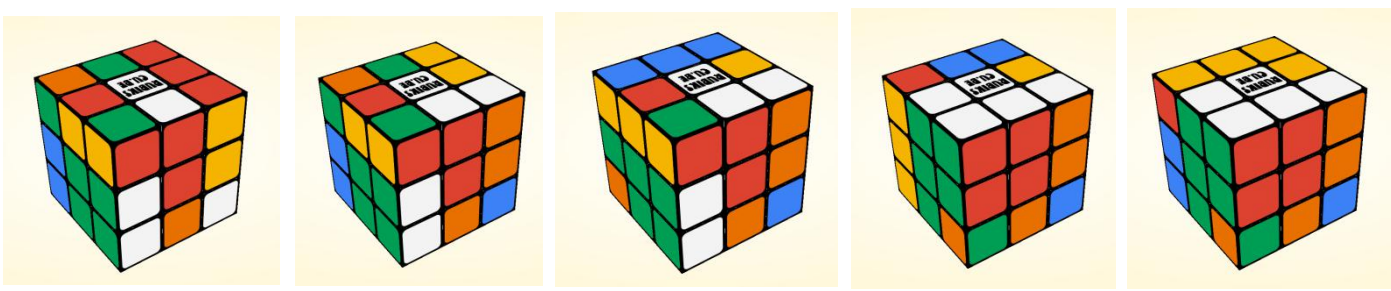


+

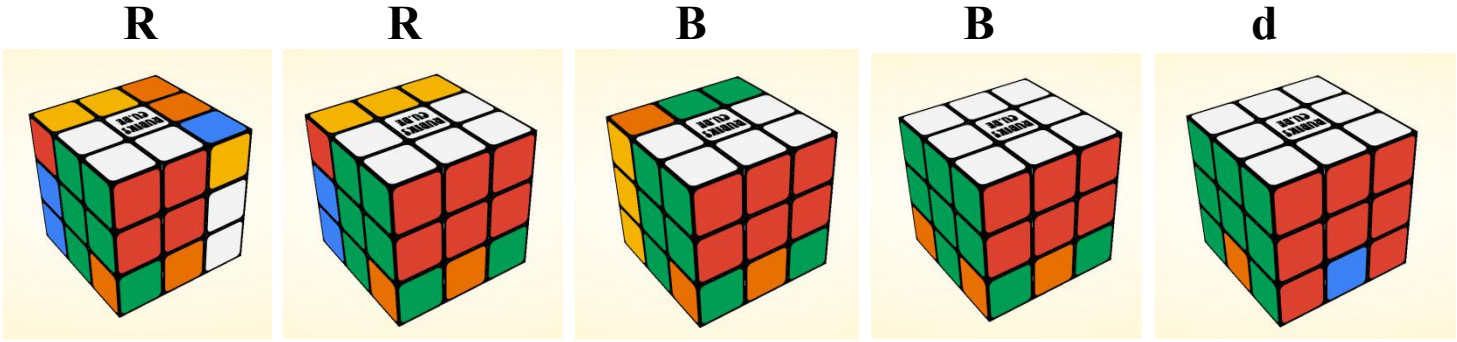
D D b L b



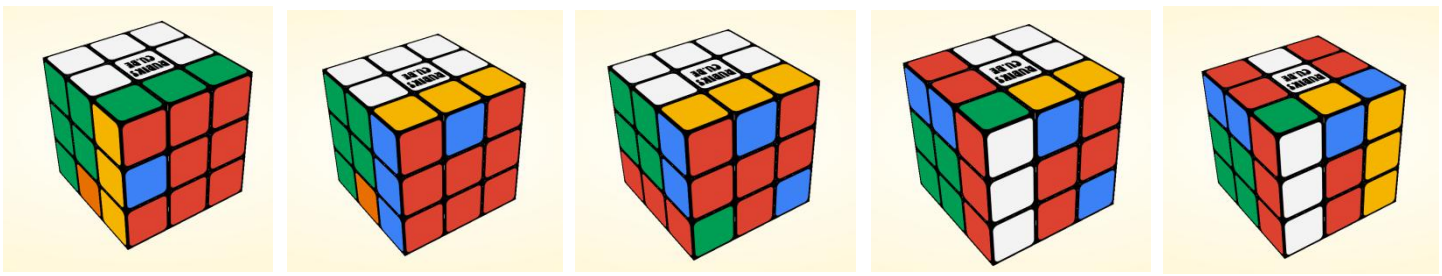
R R B l b



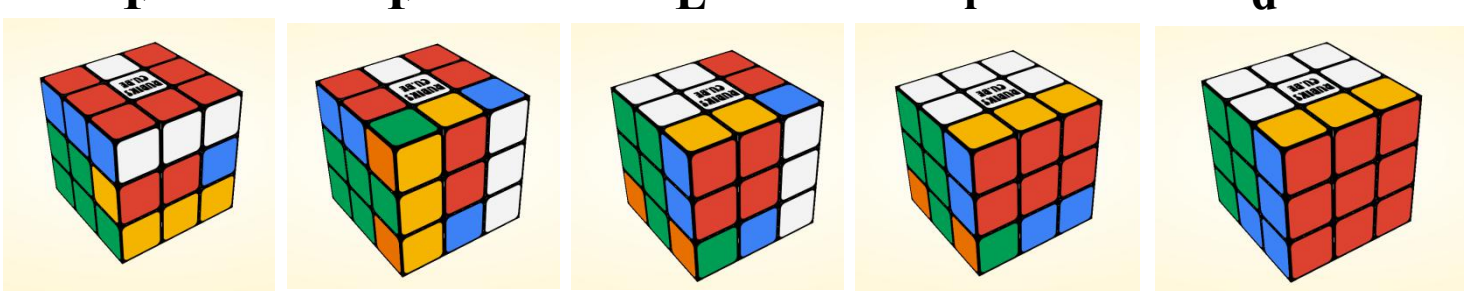
R R B B d



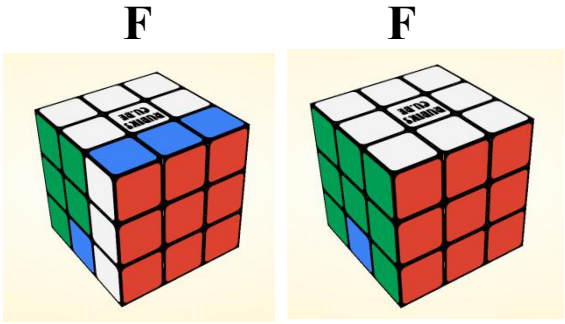
F F d l R



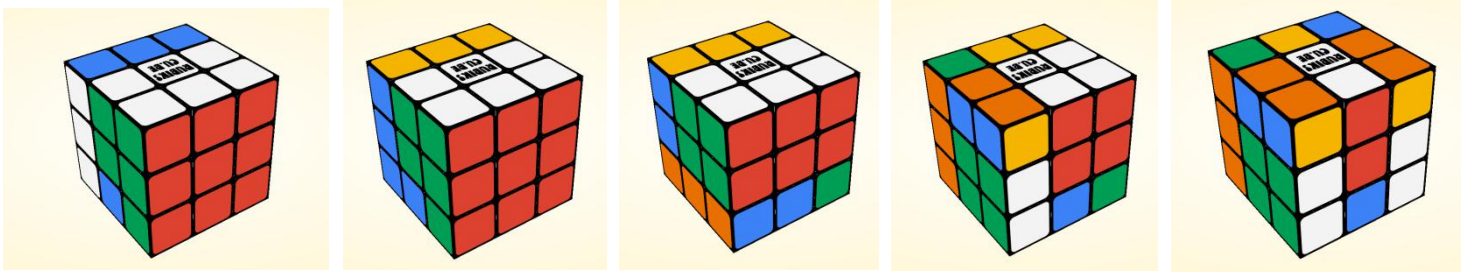
F F L r d



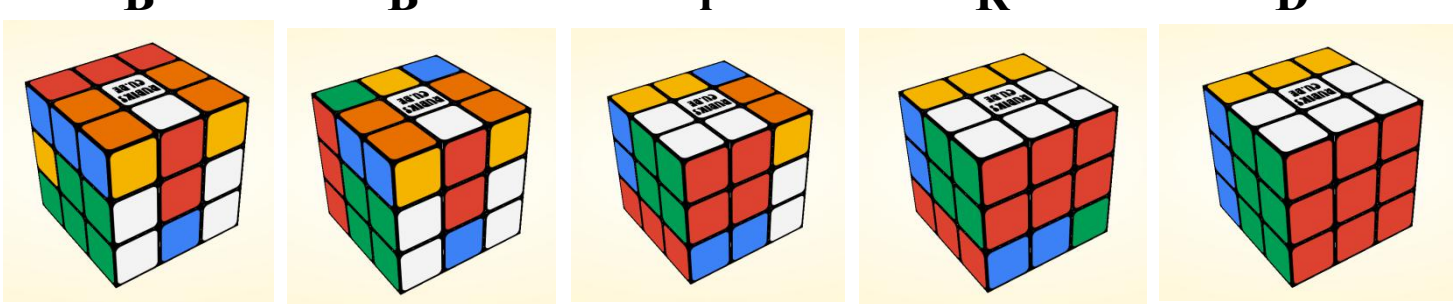
F F



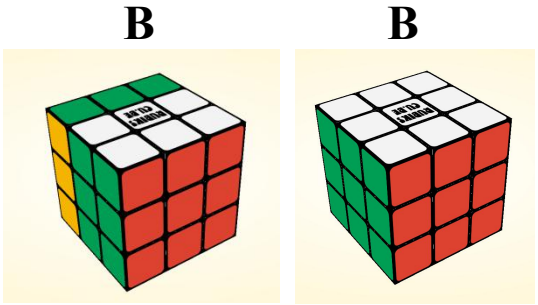
B B D L R



B B I R D



B B



So, we can see our written python code is showing correct output for a random scrambled 3x3x3 rubik's cube.

Chapter six
Result and Discussion

Describing stepper motors functionality is a broad topic but we want to describe the functionalities we have used in this project for making a rubik's cube solver machine.

In this project we have used raspberry pi as our controller for the stepper motors. There are two types of stepper motor.

1. Unipolar stepper motor
2. Bipolar stepper motor

In this project we have used bipolar stepper motors as it is programmable with a controller. Unipolar stepper motors are 5 wired or, 6 wired and bipolar stepper motors are 4 wired or, 6 wired. So, we can see in case of six wired stepper motor we need to check out its type.

As our motor driver we have used drv8825 stepper motor driver designed by pololu robotics. The drv8825 is susceptible to destructive voltage spikes. To protect the board, we should try to keep the motor power leads short and it's important to put a minimum 47 µF electrolytic capacitor across the motor power supply as close to the drv8255 as possible. In our project we have not use it and it destroyed our board and the total system after running the system successfully for a few days.

The drv8825 maximum current limit (I) is equal to twice the Vref or voltage reference.

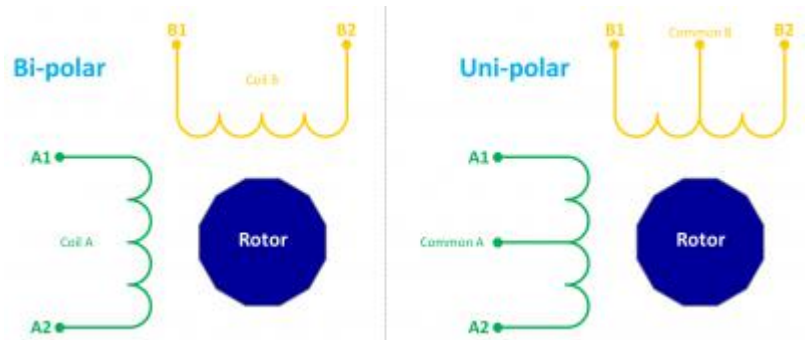
$$I_{limit} = V_{ref} \ X \ 2$$

The drv8825 datasheet indicates that in full step mode, the current is limited to 71% of the set maximum current. Therefore, we can set the maximum current higher than the peak for extra torque.

Mode2	Mode1	Mode0	Step Mode
0	0	0	Full step (2-phase excitation) with 71% current
0	0	1	1/2 step (1-2 phase excitation)
0	1	0	1/4 step (W1-2 phase excitation)
0	1	1	8 microsteps/step
1	0	0	16 microsteps/step
1	0	1	32 microsteps/step

It’s always a good idea to monitor the stepper motor and driver board temperature. 5 second rule is a good way to monitor it. If we can touch the motor and driver board for at least 5 seconds without burning our fingers then the current is probably OK. Note that the current can also be limited by the motor coil resistance because of Ohm’s law (V = I x R). For example, if our coil resistance is 30 Ω and our power supply is 12 V then the current will be limited to 0.4 A (12 V = 0.4 A x 30 Ω).

Once the maximum current is set the stepper motor can be attached. A stepper motor may have any number of coils. But these coils are connected in groups called “phases”. All the coils in a phase are energized together. 4 wire bi-polar stepper motors are 2 phase and have 2 groups of coils. The driver alternates polarity to the coils to turn the rotor. Unipolar have a center tap for each coil.



Before writing any software, we need to make sure our Pi is up to date with sudo apt-get update and sudo apt-get upgrade.

sudo apt-get update && sudo apt-get upgrade

Here, is a sample code for rotating a stepper motor’s rotor clockwise 1/4 of full circle. Here, we have used a full step (0,0,0) not to waste any amount of torque-

```
import RPi.GPIO as GPIO
from time import sleep
# for Raspberry pi

CW = 0     # Clockwise Rotation
CCW = 1    # Counterclockwise Rotation
SPR = 50  # Steps per Revolution (360 / 1.8)

# Defining pin numbers

DIR_L = 23
STEP_L = 24
MODE_L_MOTOR = (14, 15, 18)

GPIO.setmode(GPIO.BCM)
GPIO.setup(DIR_L, GPIO.OUT)
GPIO.setup(STEP_L, GPIO.OUT)
GPIO.output(DIR_L, CW)
GPIO.setup(MODE_L_MOTOR, GPIO.OUT)
GPIO.output(MODE_L_MOTOR, (0, 0, 0))

step_count = SPR
delay = .001

x=0
for x in range (step_count) :
    GPIO.output(STEP_L, GPIO.HIGH)
    sleep(delay)
    GPIO.output(STEP_L, GPIO.LOW)
    sleep(delay)
GPIO.cleanup()
```

Here, in for loop to rotate the rotor the delay is counted by (1/SPR). In our case, 360/1.8 = 200. So, 1/200 = 0.005 but in practical case it can be lower to 0.001, which is great.

Another important fact is there are some significant downsides to microstepping. There is often a substantial loss of torque which can lead to a loss of accuracy. So, we also need to be careful on that.

Through our project we have learnt a lot of things like for a high power stepper motor like Nema 23 of our project, we should use a high quality power supply which will easily provide 24 volt to all the 5 stepper motors. Another fact we have learnt is that it is an expensive project. We were concerned about it but we did not think that it will cost this much. Anyone going to do this type of Nema 23 or, raspberry pi based project, we would suggest to learn about the cost properly.

Chapter Seven
Conclusion

In robotics stepper motors are widely used. Through our project we have started our journey on learning how to run multiple stepper motors. We hope this learning will help us on future study on making real robots. And our another achievement is our python code which runs perfectly and shows full percentage of accurate output. We hope with this code running experience we will be able to write a code for bigger rubik's cube other than 3x3x3 in the future. Another achievement of our project is our newly invented method which we feel is much easier and beginner friendly and works nicely. More importantly, we want to add that we have not found any good way anywhere to solve the first 2 layers which actually inspired the idea of our project. We also have not found anywhere how to use the last algorithm properly. Then, we were able to solve the problem. Another benefit of our new technique is white center piece will face always on top. It makes the rubik's cube solving much easier. We hope all our learning through the full time of our project will take us to a long way in future study. And we also want to give thanks to our supervisor Farhin Farhad Riya for her instructions, guidance and planning which will remain in our mind for a long period of time.

Thank You.

Bibliography

Online Rubik's cube solving site :

https://rubikscu.be/

Our written python code :

https://drive.google.com/file/d/1mbowMCGvYKymLC2Qds5LnrhTEQGK97I0/view?usp=sharing

Screenshots of our full code :

https://drive.google.com/file/d/17lYwBtAhVkkuaBGdjlc1saB6RlkAwWwS/view?usp=sharing

Our research largely depends on the following websites & YouTube videos :

1. https://www.rototron.info/raspberry-pi-stepper-motor-tutorial/
2. https://www.youtube.com/watch?v=Fg8NC6CiKu0&t=2s
3. https://www.youtube.com/watch?v=VgiYULVbBTM
4. https://www.youtube.com/watch?v=ABMA3MTfm64
5. https://www.youtube.com/watch?v=UCfEuhO__K0&t=2s
6. https://www.youtube.com/watch?v=ZvGzu5ts3vM
7. https://www.youtube.com/watch?v=tNhQUrkPYJs&t=2s
8. https://www.youtube.com/watch?v=hHe4Fc6uuBs

Appendices
(Full Code)

```
import RPi.GPIO as GPIO
from time import sleep

print("\nImagine the Rubik's Cube as letter 'T' and enter the color of each piece from top most
 left side to the right and then downwards :")

asc=64
color = [[['\0' for z in range(3)]for y in range(3)] for x in range(6)]
for e in range(6):
    for f in range(3):
        for g in range(3):
            color[e][f][g]=input()

for w in range(6):
    temp=color[w][1][1]
    asc = asc + 1
    for i in range(6):
        for j in range(3):
            for k in range(3):
                if color[i][j][k]==temp:
                    color[i][j][k]=chr(asc)
# B

def B():

    temp=color[1][0][0]
    color[1][0][0]=color[2][0][0]
    color[2][0][0]=color[4][2][2]
    color[4][2][2]=color[0][0][0]
    color[0][0][0]=temp

    temp=color[1][0][1]
    color[1][0][1]=color[2][0][1]
    color[2][0][1]=color[4][2][1]
    color[4][2][1]=color[0][0][1]
    color[0][0][1]=temp

    temp=color[1][0][2]
    color[1][0][2]=color[2][0][2]
    color[2][0][2]=color[4][2][0]
    color[4][2][0]=color[0][0][2]
    color[0][0][2]=temp

    temp=color[5][0][0]
    color[5][0][0]=color[5][2][0]
    color[5][2][0]=color[5][2][2]
    color[5][2][2]=color[5][0][2]
    color[5][0][2]=temp

    temp=color[5][0][1]
    color[5][0][1]=color[5][1][0]
    color[5][1][0]=color[5][2][1]
    color[5][2][1]=color[5][1][2]
    color[5][1][2]=temp

# b

def b():
```

```
    temp=color[1][0][0]
    color[1][0][0]=color[0][0][0]
    color[0][0][0]=color[4][2][2]
    color[4][2][2]=color[2][0][0]
    color[2][0][0]=temp

    temp=color[1][0][1]
    color[1][0][1]=color[0][0][1]
    color[0][0][1]=color[4][2][1]
    color[4][2][1]=color[2][0][1]
    color[2][0][1]=temp

    temp=color[1][0][2]
    color[1][0][2]=color[0][0][2]
    color[0][0][2]=color[4][2][0]
    color[4][2][0]=color[2][0][2]
    color[2][0][2]=temp

    temp=color[5][0][0]
    color[5][0][0]=color[5][0][2]
    color[5][0][2]=color[5][2][2]
    color[5][2][2]=color[5][2][0]
    color[5][2][0]=temp

    temp=color[5][0][1]
    color[5][0][1]=color[5][1][2]
    color[5][1][2]=color[5][2][1]
    color[5][2][1]=color[5][1][0]
    color[5][1][0]=temp

# L
def L():

    temp=color[1][0][0]
    color[1][0][0]=color[5][0][0]
    color[5][0][0]=color[4][0][0]
    color[4][0][0]=color[3][0][0]
    color[3][0][0]=temp

    temp=color[1][1][0]
    color[1][1][0]=color[5][1][0]
    color[5][1][0]=color[4][1][0]
    color[4][1][0]=color[3][1][0]
    color[3][1][0]=temp

    temp=color[1][2][0]
    color[1][2][0]=color[5][2][0]
    color[5][2][0]=color[4][2][0]
    color[4][2][0]=color[3][2][0]
    color[3][2][0]=temp

    temp=color[0][0][0]
    color[0][0][0]=color[0][2][0]
    color[0][2][0]=color[0][2][2]
    color[0][2][2]=color[0][0][2]
    color[0][0][2]=temp

    temp=color[0][0][1]
    color[0][0][1]=color[0][1][0]
    color[0][1][0]=color[0][2][1]
    color[0][2][1]=color[0][1][2]
```

```
    color[0][1][2]=temp

# l

def l():

    temp=color[1][0][0]
    color[1][0][0]=color[3][0][0]
    color[3][0][0]=color[4][0][0]
    color[4][0][0]=color[5][0][0]
    color[5][0][0]=temp

    temp=color[1][1][0]
    color[1][1][0]=color[3][1][0]
    color[3][1][0]=color[4][1][0]
    color[4][1][0]=color[5][1][0]
    color[5][1][0]=temp

    temp=color[1][2][0]
    color[1][2][0]=color[3][2][0]
    color[3][2][0]=color[4][2][0]
    color[4][2][0]=color[5][2][0]
    color[5][2][0]=temp

    temp=color[0][0][0]
    color[0][0][0]=color[0][0][2]
    color[0][0][2]=color[0][2][2]
    color[0][2][2]=color[0][2][0]
    color[0][2][0]=temp

    temp=color[0][0][1]
    color[0][0][1]=color[0][1][2]
    color[0][1][2]=color[0][2][1]
    color[0][2][1]=color[0][1][0]
    color[0][1][0]=temp

# R

def R():

    temp=color[1][0][2]
    color[1][0][2]=color[3][0][2]
    color[3][0][2]=color[4][0][2]
    color[4][0][2]=color[5][0][2]
    color[5][0][2]=temp

    temp=color[1][1][2]
    color[1][1][2]=color[3][1][2]
    color[3][1][2]=color[4][1][2]
    color[4][1][2]=color[5][1][2]
    color[5][1][2]=temp

    temp=color[1][2][2]
    color[1][2][2]=color[3][2][2]
    color[3][2][2]=color[4][2][2]
    color[4][2][2]=color[5][2][2]
    color[5][2][2]=temp

    temp=color[2][0][0]
    color[2][0][0]=color[2][2][0]
    color[2][2][0]=color[2][2][2]
    color[2][2][2]=color[2][0][2]
    color[2][0][2]=temp
```

```
    temp=color[2][0][1]
    color[2][0][1]=color[2][1][0]
    color[2][1][0]=color[2][2][1]
    color[2][2][1]=color[2][1][2]
    color[2][1][2]=temp

# r

def r():

    temp=color[1][0][2]
    color[1][0][2]=color[5][0][2]
    color[5][0][2]=color[4][0][2]
    color[4][0][2]=color[3][0][2]
    color[3][0][2]=temp

    temp=color[1][1][2]
    color[1][1][2]=color[5][1][2]
    color[5][1][2]=color[4][1][2]
    color[4][1][2]=color[3][1][2]
    color[3][1][2]=temp

    temp=color[1][2][2]
    color[1][2][2]=color[5][2][2]
    color[5][2][2]=color[4][2][2]
    color[4][2][2]=color[3][2][2]
    color[3][2][2]=temp

    temp=color[2][0][0]
    color[2][0][0]=color[2][0][2]
    color[2][0][2]=color[2][2][2]
    color[2][2][2]=color[2][2][0]
    color[2][2][0]=temp

    temp=color[2][0][1]
    color[2][0][1]=color[2][1][2]
    color[2][1][2]=color[2][2][1]
    color[2][2][1]=color[2][1][0]
    color[2][1][0]=temp

# F

def F():

    temp=color[1][2][0]
    color[1][2][0]=color[0][2][0]
    color[0][2][0]=color[4][0][2]
    color[4][0][2]=color[2][2][0]
    color[2][2][0]=temp

    temp=color[1][2][1]
    color[1][2][1]=color[0][2][1]
    color[0][2][1]=color[4][0][1]
    color[4][0][1]=color[2][2][1]
    color[2][2][1]=temp

    temp=color[1][2][2]
    color[1][2][2]=color[0][2][2]
    color[0][2][2]=color[4][0][0]
    color[4][0][0]=color[2][2][2]
    color[2][2][2]=temp

    temp=color[3][0][0]
    color[3][0][0]=color[3][2][0]
```

```
    color[3][2][0]=color[3][2][2]
    color[3][2][2]=color[3][0][2]
    color[3][0][2]=temp

    temp=color[3][0][1]
    color[3][0][1]=color[3][1][0]
    color[3][1][0]=color[3][2][1]
    color[3][2][1]=color[3][1][2]
    color[3][1][2]=temp

# f

def f():

    temp=color[1][2][0]
    color[1][2][0]=color[2][2][0]
    color[2][2][0]=color[4][0][2]
    color[4][0][2]=color[0][2][0]
    color[0][2][0]=temp

    temp=color[1][2][1]
    color[1][2][1]=color[2][2][1]
    color[2][2][1]=color[4][0][1]
    color[4][0][1]=color[0][2][1]
    color[0][2][1]=temp

    temp=color[1][2][2]
    color[1][2][2]=color[2][2][2]
    color[2][2][2]=color[4][0][0]
    color[4][0][0]=color[0][2][2]
    color[0][2][2]=temp

    temp=color[3][0][0]
    color[3][0][0]=color[3][0][2]
    color[3][0][2]=color[3][2][2]
    color[3][2][2]=color[3][2][0]
    color[3][2][0]=temp

    temp=color[3][0][1]
    color[3][0][1]=color[3][1][2]
    color[3][1][2]=color[3][2][1]
    color[3][2][1]=color[3][1][0]
    color[3][1][0]=temp

# D

def D():

    temp=color[3][2][0]
    color[3][2][0]=color[0][0][0]
    color[0][0][0]=color[5][0][2]
    color[5][0][2]=color[2][2][2]
    color[2][2][2]=temp

    temp=color[3][2][1]
    color[3][2][1]=color[0][1][0]
    color[0][1][0]=color[5][0][1]
    color[5][0][1]=color[2][1][2]
    color[2][1][2]=temp

    temp=color[3][2][2]
    color[3][2][2]=color[0][2][0]
    color[0][2][0]=color[5][0][0]
```

```
    color[5][0][0]=color[2][0][2]
    color[2][0][2]=temp

    temp=color[4][0][0]
    color[4][0][0]=color[4][2][0]
    color[4][2][0]=color[4][2][2]
    color[4][2][2]=color[4][0][2]
    color[4][0][2]=temp

    temp=color[4][0][1]
    color[4][0][1]=color[4][1][0]
    color[4][1][0]=color[4][2][1]
    color[4][2][1]=color[4][1][2]
    color[4][1][2]=temp

# d

def d():

    temp=color[3][2][0]
    color[3][2][0]=color[2][2][2]
    color[2][2][2]=color[5][0][2]
    color[5][0][2]=color[0][0][0]
    color[0][0][0]=temp

    temp=color[3][2][1]
    color[3][2][1]=color[2][1][2]
    color[2][1][2]=color[5][0][1]
    color[5][0][1]=color[0][1][0]
    color[0][1][0]=temp

    temp=color[3][2][2]
    color[3][2][2]=color[2][0][2]
    color[2][0][2]=color[5][0][0]
    color[5][0][0]=color[0][2][0]
    color[0][2][0]=temp

    temp=color[4][0][0]
    color[4][0][0]=color[4][0][2]
    color[4][0][2]=color[4][2][2]
    color[4][2][2]=color[4][2][0]
    color[4][2][0]=temp

    temp=color[4][0][1]
    color[4][0][1]=color[4][1][2]
    color[4][1][2]=color[4][2][1]
    color[4][2][1]=color[4][1][0]
    color[4][1][0]=temp

# Top 4 edges

booked=[[-1 for a in range(3)]for b in range(4)]
result=[]
w=0
while w<4 :
    for x in range(6):
        for y in range(3):
            for z in range(3):

                if ((y==0 and z==1) or (y==1 and z==0) or (y==1 and z==2) or (y==2 and z==1)) a
nd color[x][y][z]==color[1][1][1] and w<4 and not [x,y,z]==booked[0] and not [x,y,z]==booked[1]
 and not [x,y,z]==booked[2]:
```

```
                    if [x,y,z]==[1,0,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,y,
z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                        B()
                        result.append('B')
                        B()
                        result.append('B')
                        if color[5][0][1]==color[5][1][1]:
                            B()
                            result.append('B')
                            B()
                            result.append('B')
                            booked[w][0]=1
                            booked[w][1]=0
                            booked[w][2]=1
                            w=w+1

                        elif color[5][0][1]==color[0][1][1]:
                            D()
                            result.append('D')
                            L()
                            result.append('L')
                            L()
                            result.append('L')

                            booked[w][0]=1
                            booked[w][1]=1
                            booked[w][2]=0
                            w=w+1

                        elif color[5][0][1]==color[2][1][1]:
                            d()
                            result.append('d')
                            R()
                            result.append('R')
                            R()
                            result.append('R')
                            booked[w][0]=1
                            booked[w][1]=1
                            booked[w][2]=2
                            w=w+1

                        elif color[5][0][1]==color[3][1][1]:
                            D()
                            result.append('D')
                            D()
                            result.append('D')
                            F()
                            result.append('F')
                            F()
                            result.append('F')

                            booked[w][0]=1
                            booked[w][1]=2
                            booked[w][2]=1
                            w=w+1

                    elif [x,y,z]==[1,1,0] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                        L()
                        result.append('L')
                        L()
                        result.append('L')
```

```
if color[0][1][0]==color[0][1][1]:
    L()
    result.append('L')
    L()
    result.append('L')

    booked[w][0]=1
    booked[w][1]=1
    booked[w][2]=0
    w=w+1

elif color[0][1][0]==color[3][1][1]:
    D()
    result.append('D')
    F()
    result.append('F')
    F()
    result.append('F')

    booked[w][0]=1
    booked[w][1]=2
    booked[w][2]=1
    w=w+1

elif color[0][1][0]==color[5][1][1]:
    d()
    result.append('d')
    B()
    result.append('B')
    B()
    result.append('B')
                    booked[w][0]=1
                    booked[w][1]=0
                    booked[w][2]=1
                    w=w+1


                elif color[0][1][0]==color[2][1][1]:
                    D()
                    result.append('D')
                    D()
                    result.append('D')
                    R()
                    result.append('R')
                    R()
                    result.append('R')

                    booked[w][0]=1
                    booked[w][1]=1
                    booked[w][2]=2
                    w=w+1

                elif [x,y,z]==[1,1,2] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    R()
                    result.append('R')
                    R()
                    result.append('R')
                    if color[2][1][2]==color[2][1][1]:
                        R()
                        result.append('R')
                        R()
```

```
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[2][1][2]==color[5][1][1]:
            D()
            result.append('D')
            B()
            result.append('B')
            B()
            result.append('B')
            booked[w][0]=1
            booked[w][1]=0
            booked[w][2]=1
            w=w+1

        elif color[2][1][2]==color[3][1][1]:
            d()
            result.append('d')
            F()
            result.append('F')
            F()
            result.append('F')
            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[2][1][2]==color[0][1][1]:
                    D()
                    result.append('D')
                    D()
                    result.append('D')
                    L()
                    result.append('L')
                    L()
                    result.append('L')

                    booked[w][0]=1
                    booked[w][1]=1
                    booked[w][2]=0
                    w=w+1

                elif [x,y,z]==[1,2,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    F()
                    result.append('F')
                    F()
                    result.append('F')
                    if color[3][2][1]==color[3][1][1]:

                        F()
                        result.append('F')
                        F()
                        result.append('F')

                        booked[w][0]=1
                        booked[w][1]=2
                        booked[w][2]=1
```

```
        w=w+1

    elif color[3][2][1]==color[2][1][1]:

        D()
        result.append('D')
        R()
        result.append('R')
        R()
        result.append('R')
        booked[w][0]=1
        booked[w][1]=1
        booked[w][2]=2
        w=w+1

    elif color[3][2][1]==color[0][1][1]:
        d()
        result.append('d')
        L()
        result.append('L')
        L()
        result.append('L')

        booked[w][0]=1
        booked[w][1]=1
        booked[w][2]=0
        w=w+1

    elif color[3][2][1]==color[5][1][1]:
        D()
        result.append('D')
                                D()
                                result.append('D')

                                B()
                                result.append('B')
                                B()
                                result.append('B')
                                booked[w][0]=1
                                booked[w][1]=0
                                booked[w][2]=1
                                w=w+1

                        elif [x,y,z]==[5,2,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                            B()
                            result.append('B')
                            l()
                            result.append('l')
                            d()
                            result.append('d')
                            L()
                            result.append('L')
                            D()
                            result.append('D')
                            b()
                            result.append('b')
                            if color[0][1][0]==color[0][1][1]:
                                L()
                                result.append('L')
                                L()
                                result.append('L')
```

```
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[0][1][0]==color[3][1][1]:
            D()
            result.append('D')
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[5][1][1]:
            d()
            result.append('d')
            B()
            result.append('B')
            B()
            result.append('B')
            booked[w][0]=1
            booked[w][1]=0
            booked[w][2]=1
            w=w+1

                elif color[0][1][0]==color[2][1][1]:
                    D()
                    result.append('D')
                    D()
                    result.append('D')
                    R()
                    result.append('R')
                    R()
                    result.append('R')

                    booked[w][0]=1
                    booked[w][1]=1
                    booked[w][2]=2
                    w=w+1

                elif [x,y,z]==[5,1,2] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    R()
                    result.append('R')
                    D()
                    result.append('D')
                    r()
                    result.append('r')
                    if color[5][0][1]==color[5][1][1]:
                        B()
                        result.append('B')
                        B()
                        result.append('B')
                        booked[w][0]=1
                        booked[w][1]=0
                        booked[w][2]=1
```

```
            w=w+1

        elif color[5][0][1]==color[2][1][1]:
            d()
            result.append('d')
            R()
            result.append('R')
            R()
            result.append('R')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[5][0][1]==color[0][1][1]:
            D()
            result.append('D')
            L()
            result.append('L')
            L()
            result.append('L')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[5][0][1]==color[3][1][1]:
            D()
            result.append('D')
            D()
            result.append('D')
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

                elif [x,y,z]==[5,1,0] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    l()
                    result.append('l')
                    d()
                    result.append('d')
                    L()
                    result.append('L')
                    if color[5][0][1]==color[5][1][1]:
                        B()
                        result.append('B')
                        B()
                        result.append('B')
                        booked[w][0]=1
                        booked[w][1]=0
                        booked[w][2]=1
                        w=w+1

                    elif color[5][0][1]==color[2][1][1]:
                        d()
```

```
            result.append('d')
            R()
            result.append('R')
            R()
            result.append('R')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[5][0][1]==color[0][1][1]:
            D()
            result.append('D')
            L()
            result.append('L')
            L()
            result.append('L')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[5][0][1]==color[3][1][1]:
            D()
            result.append('D')
            D()
            result.append('D')
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

                elif [x,y,z]==[5,0,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    B()
                    result.append('B')
                    R()
                    result.append('R')
                    D()
                    result.append('D')
                    r()
                    result.append('r')
                    d()
                    result.append('d')
                    b()
                    result.append('b')
                    if color[2][1][2]==color[2][1][1]:
                        R()
                        result.append('R')
                        R()
                        result.append('R')
                        booked[w][0]=1
                        booked[w][1]=1
                        booked[w][2]=2
                        w=w+1
```

```
elif color[2][1][2]==color[5][1][1]:
    D()
    result.append('D')
    B()
    result.append('B')
    B()
    result.append('B')
    booked[w][0]=1
    booked[w][1]=0
    booked[w][2]=1
    w=w+1

elif color[2][1][2]==color[3][1][1]:
    d()
    result.append('d')
    F()
    result.append('F')
    F()
    result.append('F')
    booked[w][0]=1
    booked[w][1]=2
    booked[w][2]=1
    w=w+1

elif color[2][1][2]==color[0][1][1]:
    D()
    result.append('D')
    D()
    result.append('D')
    L()
    result.append('L')
    L()
    result.append('L')

    booked[w][0]=1
    booked[w][1]=1
    booked[w][2]=0
    w=w+1

                elif [x,y,z]==[0,1,2] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    L()
                    result.append('L')
                    f()
                    result.append('f')
                    d()
                    result.append('d')
                    F()
                    result.append('F')
                    D()
                    result.append('D')
                    l()
                    result.append('l')

                    if color[3][2][1]==color[3][1][1]:

                        F()
                        result.append('F')
                        F()
                        result.append('F')

                        booked[w][0]=1
```

```
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[3][2][1]==color[2][1][1]:

            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[3][2][1]==color[0][1][1]:
            d()
            result.append('d')
            L()
            result.append('L')
            L()
            result.append('L')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[3][2][1]==color[5][1][1]:
                            D()
                            result.append('D')
                            D()
                            result.append('D')

                            B()
                            result.append('B')
                            B()
                            result.append('B')
                            booked[w][0]=1
                            booked[w][1]=0
                            booked[w][2]=1
                            w=w+1

                    elif [x,y,z]==[0,0,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                            B()
                            result.append('B')
                            D()
                            result.append('D')
                            b()
                            result.append('b')

                            if color[0][1][0]==color[0][1][1]:
                                L()
                                result.append('L')
                                L()
                                result.append('L')

                                booked[w][0]=1
                                booked[w][1]=1
```

```
            booked[w][2]=0
            w=w+1

        elif color[0][1][0]==color[3][1][1]:
            D()
            result.append('D')
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[5][1][1]:
            d()
            result.append('d')
            B()
            result.append('B')
            B()
            result.append('B')
            booked[w][0]=1
            booked[w][1]=0
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[2][1][1]:
            D()
            result.append('D')
            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

                elif [x,y,z]==[0,2,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    f()
                    result.append('f')
                    d()
                    result.append('d')
                    F()
                    result.append('F')
                    if color[0][1][0]==color[0][1][1]:
                        L()
                        result.append('L')
                        L()
                        result.append('L')

                        booked[w][0]=1
                        booked[w][1]=1
                        booked[w][2]=0
                        w=w+1
```

```
elif color[0][1][0]==color[3][1][1]:
    D()
    result.append('D')
    F()
    result.append('F')
    F()
    result.append('F')

    booked[w][0]=1
    booked[w][1]=2
    booked[w][2]=1
    w=w+1

elif color[0][1][0]==color[5][1][1]:
    d()
    result.append('d')
    B()
    result.append('B')
    B()
    result.append('B')
    booked[w][0]=1
    booked[w][1]=0
    booked[w][2]=1
    w=w+1

elif color[0][1][0]==color[2][1][1]:
    D()
    result.append('D')
    D()
    result.append('D')
    R()
    result.append('R')
    R()
    result.append('R')

    booked[w][0]=1
    booked[w][1]=1
    booked[w][2]=2
    w=w+1

elif [x,y,z]==[0,1,0] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
    l()
    result.append('l')
    f()
    result.append('f')
    d()
    result.append('d')
    F()
    result.append('F')
    D()
    result.append('D')
    L()
    result.append('L')
    if color[3][2][1]==color[3][1][1]:

        F()
        result.append('F')
        F()
        result.append('F')

        booked[w][0]=1
```

```
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[3][2][1]==color[2][1][1]:

            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[3][2][1]==color[0][1][1]:
            d()
            result.append('d')
            L()
            result.append('L')
            L()
            result.append('L')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[3][2][1]==color[5][1][1]:
                            D()
                            result.append('D')
                            D()
                            result.append('D')

                            B()
                            result.append('B')
                            B()
                            result.append('B')
                            booked[w][0]=1
                            booked[w][1]=0
                            booked[w][2]=1
                            w=w+1

                    elif [x,y,z]==[2,1,0] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                            r()
                            result.append('r')
                            F()
                            result.append('F')
                            D()
                            result.append('D')
                            f()
                            result.append('f')
                            d()
                            result.append('d')
                            R()
                            result.append('R')

                            if color[3][2][1]==color[3][1][1]:
```

```
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[3][2][1]==color[2][1][1]:

            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[3][2][1]==color[0][1][1]:
            d()
            result.append('d')
            L()
            result.append('L')
            L()
            result.append('L')
                booked[w][0]=1
                booked[w][1]=1
                booked[w][2]=0
                w=w+1

            elif color[3][2][1]==color[5][1][1]:
                D()
                result.append('D')
                D()
                result.append('D')

                B()
                result.append('B')
                B()
                result.append('B')
                booked[w][0]=1
                booked[w][1]=0
                booked[w][2]=1
                w=w+1

                elif [x,y,z]==[2,2,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    F()
                    result.append('F')
                    d()
                    result.append('d')
                    f()
                    result.append('f')
                    if color[0][1][0]==color[0][1][1]:
                        L()
                        result.append('L')
```

```
            L()
            result.append('L')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[0][1][0]==color[3][1][1]:
            D()
            result.append('D')
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[5][1][1]:
            d()
            result.append('d')
            B()
            result.append('B')
            B()
            result.append('B')
            booked[w][0]=1
            booked[w][1]=0
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[2][1][1]:
            D()
            result.append('D')
            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

    elif [x,y,z]==[2,0,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
            b()
            result.append('b')
            D()
            result.append('D')
            B()
            result.append('B')
            if color[0][1][0]==color[0][1][1]:
                L()
                result.append('L')
                L()
                result.append('L')
```

```
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[0][1][0]==color[3][1][1]:
            D()
            result.append('D')
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[5][1][1]:
            d()
            result.append('d')
            B()
            result.append('B')
            B()
            result.append('B')
            booked[w][0]=1
            booked[w][1]=0
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[2][1][1]:
            D()
            result.append('D')
            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1


                elif [x,y,z]==[2,1,2] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    R()
                    result.append('R')
                    F()
                    result.append('F')
                    D()
                    result.append('D')
                    f()
                    result.append('f')
                    d()
                    result.append('d')
                    r()
                    result.append('r')

                    if color[3][2][1]==color[3][1][1]:
```

```
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[3][2][1]==color[2][1][1]:

            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[3][2][1]==color[0][1][1]:
            d()
            result.append('d')
            L()
            result.append('L')
            L()
            result.append('L')
                booked[w][0]=1
                booked[w][1]=1
                booked[w][2]=0
                w=w+1

            elif color[3][2][1]==color[5][1][1]:
                D()
                result.append('D')
                D()
                result.append('D')

                B()
                result.append('B')
                B()
                result.append('B')
                booked[w][0]=1
                booked[w][1]=0
                booked[w][2]=1
                w=w+1

                elif [x,y,z]==[3,0,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    F()
                    result.append('F')
                    r()
                    result.append('r')
                    d()
                    result.append('d')
                    R()
                    result.append('R')
```

```
D()
result.append('D')
f()
result.append('f')
if color[2][1][2]==color[2][1][1]:
    R()
    result.append('R')
    R()
    result.append('R')
    booked[w][0]=1
    booked[w][1]=1
    booked[w][2]=2
    w=w+1

elif color[2][1][2]==color[5][1][1]:
    D()
    result.append('D')
    B()
    result.append('B')
    B()
    result.append('B')
    booked[w][0]=1
    booked[w][1]=0
    booked[w][2]=1
    w=w+1

elif color[2][1][2]==color[3][1][1]:
    d()
    result.append('d')
    F()
    result.append('F')
    booked[w][0]=1
    booked[w][1]=2
    booked[w][2]=1
    w=w+1

elif color[2][1][2]==color[0][1][1]:
    D()
    result.append('D')
    D()
    result.append('D')
    L()
    result.append('L')
    L()
    result.append('L')

    booked[w][0]=1
    booked[w][1]=1
    booked[w][2]=0
    w=w+1

elif [x,y,z]==[3,1,0] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
    L()
    result.append('L')
    D()
    result.append('D')
    l()
    result.append('l')
    if color[3][2][1]==color[3][1][1]:
```

```
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[3][2][1]==color[2][1][1]:

            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[3][2][1]==color[0][1][1]:
            d()
            result.append('d')
            L()
            result.append('L')
            L()
            result.append('L')
                booked[w][0]=1
                booked[w][1]=1
                booked[w][2]=0
                w=w+1

            elif color[3][2][1]==color[5][1][1]:
                D()
                result.append('D')
                D()
                result.append('D')
                B()
                result.append('B')
                B()
                result.append('B')
                booked[w][0]=1
                booked[w][1]=0
                booked[w][2]=1
                w=w+1

                elif [x,y,z]==[3,1,2] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    r()
                    result.append('r')
                    d()
                    result.append('d')
                    R()
                    result.append('R')

                    if color[3][2][1]==color[3][1][1]:
```

```
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[3][2][1]==color[2][1][1]:

            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[3][2][1]==color[0][1][1]:
            d()
            result.append('d')
            L()
            result.append('L')
            L()
            result.append('L')
                booked[w][0]=1
                booked[w][1]=1
                booked[w][2]=0
                w=w+1

            elif color[3][2][1]==color[5][1][1]:
                D()
                result.append('D')
                D()
                result.append('D')

                B()
                result.append('B')
                B()
                result.append('B')
                booked[w][0]=1
                booked[w][1]=0
                booked[w][2]=1
                w=w+1

                elif [x,y,z]==[3,2,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    F()
                    result.append('F')
                    L()
                    result.append('L')
                    D()
                    result.append('D')
                    l()
                    result.append('l')
                    d()
```

```
result.append('d')
f()
result.append('f')
if color[0][1][0]==color[0][1][1]:
    L()
    result.append('L')
    L()
    result.append('L')

    booked[w][0]=1
    booked[w][1]=1
    booked[w][2]=0
    w=w+1

elif color[0][1][0]==color[3][1][1]:
    D()
    result.append('D')
    F()
    result.append('F')
    F()
    result.append('F')

    booked[w][0]=1
    booked[w][1]=2
    booked[w][2]=1
    w=w+1

elif color[0][1][0]==color[5][1][1]:
    d()
    result.append('d')
    B()
    result.append('B')
    B()
    result.append('B')
    booked[w][0]=1
    booked[w][1]=0
    booked[w][2]=1
    w=w+1


elif color[0][1][0]==color[2][1][1]:
    D()
    result.append('D')
    D()
    result.append('D')
    R()
    result.append('R')
    R()
    result.append('R')

    booked[w][0]=1
    booked[w][1]=1
    booked[w][2]=2
    w=w+1

elif [x,y,z]==[4,0,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
    if color[3][2][1]==color[3][1][1]:

        F()
        result.append('F')
        F()
```

```
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[3][2][1]==color[2][1][1]:

            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')
            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

        elif color[3][2][1]==color[0][1][1]:
            d()
            result.append('d')
            L()
            result.append('L')
            L()
            result.append('L')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[3][2][1]==color[5][1][1]:
            D()
            result.append('D')
            D()
            result.append('D')

            B()
            result.append('B')
            B()
            result.append('B')
            booked[w][0]=1
            booked[w][1]=0
            booked[w][2]=1
            w=w+1

    elif [x,y,z]==[4,1,0] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
        if color[0][1][0]==color[0][1][1]:
            L()
            result.append('L')
            L()
            result.append('L')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

        elif color[0][1][0]==color[3][1][1]:
```

```
            D()
            result.append('D')
            F()
            result.append('F')
            F()
            result.append('F')

            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[0][1][0]==color[5][1][1]:
            d()
            result.append('d')
            B()
            result.append('B')
            B()
            result.append('B')
            booked[w][0]=1
            booked[w][1]=0
            booked[w][2]=1
            w=w+1


        elif color[0][1][0]==color[2][1][1]:
            D()
            result.append('D')
            D()
            result.append('D')
            R()
            result.append('R')
            R()
            result.append('R')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=2
            w=w+1

                elif [x,y,z]==[4,1,2] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                    if color[2][1][2]==color[2][1][1]:
                        R()
                        result.append('R')
                        R()
                        result.append('R')
                        booked[w][0]=1
                        booked[w][1]=1
                        booked[w][2]=2
                        w=w+1

                    elif color[2][1][2]==color[5][1][1]:
                        D()
                        result.append('D')
                        B()
                        result.append('B')
                        B()
                        result.append('B')
                        booked[w][0]=1
                        booked[w][1]=0
                        booked[w][2]=1
```

```
            w=w+1

        elif color[2][1][2]==color[3][1][1]:
            d()
            result.append('d')
            F()
            result.append('F')
            F()
            result.append('F')
            booked[w][0]=1
            booked[w][1]=2
            booked[w][2]=1
            w=w+1

        elif color[2][1][2]==color[0][1][1]:

            D()
            result.append('D')
            D()
            result.append('D')
            L()
            result.append('L')
            L()
            result.append('L')

            booked[w][0]=1
            booked[w][1]=1
            booked[w][2]=0
            w=w+1

    elif [x,y,z]==[4,2,1] and color[x][y][z]==color[1][1][1] and w<4 and not [x,
y,z]==booked[0] and not [x,y,z]==booked[1] and not [x,y,z]==booked[2]:
                if color[5][0][1]==color[5][1][1]:
                    B()
                    result.append('B')
                    B()
                    result.append('B')
                    booked[w][0]=1
                    booked[w][1]=0
                    booked[w][2]=1
                    w=w+1

                elif color[5][0][1]==color[2][1][1]:
                    d()
                    result.append('d')
                    R()
                    result.append('R')
                    R()
                    result.append('R')

                    booked[w][0]=1
                    booked[w][1]=1
                    booked[w][2]=2
                    w=w+1

                elif color[5][0][1]==color[0][1][1]:
                    D()
                    result.append('D')
                    L()
                    result.append('L')
                    L()
                    result.append('L')
```

```
                booked[w][0]=1
                booked[w][1]=1
                booked[w][2]=0
                w=w+1

            elif color[5][0][1]==color[3][1][1]:
                D()
                result.append('D')
                D()
                result.append('D')
                F()
                result.append('F')
                F()
                result.append('F')

                booked[w][0]=1
                booked[w][1]=2
                booked[w][2]=1
                w=w+1


result.append('+')

# 2nd layer 4 edges

if not (color[0][0][1]==color[0][1][1] and color[5][1][0]==color[5][1][1]):
    if color[0][0][1]==color[5][1][1] and color[5][1][0]==color[0][1][1]:
        B()
        result.append('B')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
        l()
        result.append('l')
        D()
        result.append('D')
        L()
        result.append('L')
    elif color[0][2][1]==color[0][1][1] and color[3][1][0]==color[5][1][1]:
        f()
        result.append('f')
        B()
        result.append('B')
        D()
        result.append('D')
        D()
        result.append('D')
        b()
        result.append('b')
        F()
        result.append('F')
    elif color[0][2][1]==color[5][1][1] and color[3][1][0]==color[0][1][1]:
        L()
        result.append('L')
        d()
        result.append('d')
        l()
        result.append('l')
        D()
```

```
        result.append('D')
        B()
        result.append('B')
        d()
        result.append('d')
        b()
        result.append('b')
    elif color[0][1][0]==color[0][1][1] and color[4][1][0]==color[5][1][1]:
        d()
        result.append('d')
        l()
        result.append('l')
        D()
        result.append('D')
        L()
        result.append('L')
    elif color[0][1][0]==color[5][1][1] and color[4][1][0]==color[0][1][1]:
        B()
        result.append('B')
        d()
        result.append('d')
        b()
        result.append('b')
    elif color[2][0][1]==color[0][1][1] and color[5][1][2]==color[5][1][1]:
        R()
        result.append('R')
        l()
        result.append('l')
        D()
        result.append('D')
        D()
        result.append('D')
        L()
        result.append('L')
        r()
        result.append('r')
    elif color[2][0][1]==color[5][1][1] and color[5][1][2]==color[0][1][1]:
        b()
        result.append('b')
        D()
        result.append('D')
        B()
        result.append('B')
        d()
        result.append('d')
        l()
        result.append('l')
        D()
        result.append('D')
        L()
        result.append('L')
    elif color[2][2][1]==color[0][1][1] and color[3][1][2]==color[5][1][1]:
        F()
        result.append('F')
        B()
        result.append('B')
        D()
        result.append('D')
        D()
        result.append('D')
        b()
        result.append('b')
```

```
        f()
        result.append('f')
    elif color[2][2][1]==color[5][1][1] and color[3][1][2]==color[0][1][1]:
        r()
        result.append('r')
        B()
        result.append('B')
        D()
        result.append('D')
        b()
        result.append('b')
        R()
        result.append('R')
    elif color[2][1][2]==color[0][1][1] and color[4][1][2]==color[5][1][1]:
        l()
        result.append('l')
        D()
        result.append('D')
        D()
        result.append('D')
        L()
        result.append('L')
    elif color[2][1][2]==color[5][1][1] and color[4][1][2]==color[0][1][1]:
        B()
        result.append('B')
        D()
        result.append('D')
        b()
        result.append('b')
    elif color[3][2][1]==color[0][1][1] and color[4][0][1]==color[5][1][1]:
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
    elif color[3][2][1]==color[5][1][1] and color[4][0][1]==color[0][1][1]:
        B()
        result.append('B')
        D()
        result.append('D')
        D()
        result.append('D')
        b()
        result.append('b')
    elif color[5][0][1]==color[0][1][1] and color[4][2][1]==color[5][1][1]:
        l()
        result.append('l')
        D()
        result.append('D')
        L()
        result.append('L')
    elif color[5][0][1]==color[5][1][1] and color[4][2][1]==color[0][1][1]:
        D()
        result.append('D')
        B()
        result.append('B')
        d()
        result.append('d')
        b()
        result.append('b')
```

```
result.append('+')

if not (color[0][2][1]==color[0][1][1] and color[3][1][0]==color[3][1][1]):
    if color[0][2][1]==color[3][1][1] and color[3][1][0]==color[0][1][1]:
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
        L()
        result.append('L')
        d()
        result.append('d')
        l()
        result.append('l')
    elif  color[0][0][1]==color[0][1][1] and color[5][1][0]==color[3][1][1]:
        B()
        result.append('B')
        f()
        result.append('f')
        D()
        result.append('D')
        D()
        result.append('D')
        F()
        result.append('F')
        b()
        result.append('b')
elif color[0][0][1]==color[3][1][1] and color[5][1][0]==color[0][1][1]:
    l()
    result.append('l')
    D()
    result.append('D')
    L()
    result.append('L')
    d()
    result.append('d')
    f()
    result.append('f')
    D()
    result.append('D')
    F()
    result.append('F')
elif color[0][1][0]==color[0][1][1] and color[4][1][0]==color[3][1][1]:
    D()
    result.append('D')
    L()
    result.append('L')
    d()
    result.append('d')
    l()
    result.append('l')
elif color[0][1][0]==color[3][1][1] and color[4][1][0]==color[0][1][1]:
    f()
    result.append('f')
    D()
    result.append('D')
    F()
    result.append('F')
```

```
elif color[2][0][1]==color[0][1][1] and color[5][1][2]==color[3][1][1]:
    b()
    result.append('b')
    f()
    result.append('f')
    D()
    result.append('D')
    D()
    result.append('D')
    F()
    result.append('F')
    B()
    result.append('B')
elif color[2][0][1]==color[3][1][1] and color[5][1][2]==color[0][1][1]:
    b()
    result.append('b')
    D()
    result.append('D')
    D()
    result.append('D')
    B()
    result.append('B')
    L()
    result.append('L')
    d()
    result.append('d')
    l()
    result.append('l')
elif color[2][2][1]==color[0][1][1] and color[3][1][2]==color[3][1][1]:
    r()
    result.append('r')
    L()
    result.append('L')
    D()
    result.append('D')
    D()
    result.append('D')
    l()
    result.append('l')
    R()
    result.append('R')
elif color[2][2][1]==color[3][1][1] and color[3][1][2]==color[0][1][1]:
    F()
    result.append('F')
    d()
    result.append('d')
    f()
    result.append('f')
    D()
    result.append('D')
    L()
    result.append('L')
    d()
    result.append('d')
    l()
    result.append('l')
elif color[2][1][2]==color[0][1][1] and color[4][1][2]==color[3][1][1]:
    L()
    result.append('L')
    D()
    result.append('D')
    D()
```

```
        result.append('D')
        l()
        result.append('l')
    elif color[2][1][2]==color[3][1][1] and color[4][1][2]==color[0][1][1]:
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
    elif color[3][2][1]==color[0][1][1] and color[4][0][1]==color[3][1][1]:
        L()
        result.append('L')
        d()
        result.append('d')
        l()
        result.append('l')
    elif color[3][2][1]==color[3][1][1] and color[4][0][1]==color[0][1][1]:
        d()
        result.append('d')
        f()
        result.append('f')
        D()
        result.append('D')
        F()
        result.append('F')
    elif color[5][0][1]==color[0][1][1] and color[4][2][1]==color[3][1][1]:
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
    elif color[5][0][1]==color[3][1][1] and color[4][2][1]==color[0][1][1]:
        f()
        result.append('f')
        D()
        result.append('D')
        D()
        result.append('D')
        F()
        result.append('F')

result.append('+')

if not (color[2][0][1]==color[2][1][1] and color[5][1][2]==color[5][1][1]):
    if color[2][0][1]==color[5][1][1] and color[5][1][2]==color[2][1][1]:
        R()
        result.append('R')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
        b()
        result.append('b')
        D()
        result.append('D')
        B()
        result.append('B')
    elif color[2][2][1]==color[2][1][1] and color[3][1][2]==color[5][1][1]:
```

```
        F()
        result.append('F')
        b()
        result.append('b')
        D()
        result.append('D')
        D()
        result.append('D')
        B()
        result.append('B')
        f()
        result.append('f')
    elif color[2][2][1]==color[5][1][1] and color[3][1][2]==color[2][1][1]:
        r()
        result.append('r')
        D()
        result.append('D')
        R()
        result.append('R')
        d()
        result.append('d')
        b()
        result.append('b')
        D()
        result.append('D')
        B()
        result.append('B')
    elif color[2][1][2]==color[2][1][1] and color[4][1][2]==color[5][1][1]:
        D()
        result.append('D')
        R()
        result.append('R')
        d()
        result.append('d')
        r()
        result.append('r')
    elif color[2][1][2]==color[5][1][1] and color[4][1][2]==color[2][1][1]:
        b()
        result.append('b')
        D()
        result.append('D')
        B()
        result.append('B')
    elif color[0][0][1]==color[2][1][1] and color[5][1][0]==color[5][1][1]:
        l()
        result.append('l')
        R()
        result.append('R')
        D()
        result.append('D')
        D()
        result.append('D')
        r()
        result.append('r')
        L()
        result.append('L')
    elif color[0][0][1]==color[5][1][1] and color[5][1][0]==color[2][1][1]:
        B()
        result.append('B')
        d()
        result.append('d')
        b()
```

```
        result.append('b')
        D()
        result.append('D')
        R()
        result.append('R')
        d()
        result.append('d')
        r()
        result.append('r')
    elif color[0][2][1]==color[2][1][1] and color[3][1][0]==color[5][1][1]:
        f()
        result.append('f')
        b()
        result.append('b')
        D()
        result.append('D')
        D()
        result.append('D')
        B()
        result.append('B')
        F()
        result.append('F')
    elif color[0][2][1]==color[5][1][1] and color[3][1][0]==color[2][1][1]:
        L()
        result.append('L')
        D()
        result.append('D')
        D()
        result.append('D')
        l()
        result.append('l')
        b()
        result.append('b')
        D()
        result.append('D')
        B()
        result.append('B')
    elif color[0][1][0]==color[2][1][1] and color[4][1][0]==color[5][1][1]:
        R()
        result.append('R')
        D()
        result.append('D')
        D()
        result.append('D')
        r()
        result.append('r')
    elif color[0][1][0]==color[5][1][1] and color[4][1][0]==color[2][1][1]:
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
    elif color[3][2][1]==color[2][1][1] and color[4][0][1]==color[5][1][1]:
        R()
        result.append('R')
        D()
        result.append('D')
        r()
        result.append('r')
    elif color[3][2][1]==color[5][1][1] and color[4][0][1]==color[2][1][1]:
        b()
```

```
            result.append('b')
            D()
            result.append('D')
            D()
            result.append('D')
            B()
            result.append('B')
        elif color[5][0][1]==color[2][1][1] and color[4][2][1]==color[5][1][1]:
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
        elif color[5][0][1]==color[5][1][1] and color[4][2][1]==color[2][1][1]:
            d()
            result.append('d')
            b()
            result.append('b')
            D()
            result.append('D')
            B()
            result.append('B')

    result.append('+')

    if not (color[2][2][1]==color[2][1][1] and color[3][1][2]==color[3][1][1]):
        if color[2][2][1]==color[3][1][1] and color[3][1][2]==color[2][1][1]:
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
            r()
            result.append('r')
            D()
            result.append('D')
            R()
            result.append('R')
        elif color[2][0][1]==color[2][1][1] and color[5][1][2]==color[3][1][1]:
            b()
            result.append('b')
            F()
            result.append('F')
            D()
            result.append('D')
            D()
            result.append('D')
            f()
            result.append('f')
            B()
            result.append('B')
        elif color[2][0][1]==color[3][1][1] and color[5][1][2]==color[2][1][1]:
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
```

```
        D()
        result.append('D')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
    elif color[2][1][2]==color[2][1][1] and color[4][1][2]==color[3][1][1]:
        d()
        result.append('d')
        r()
        result.append('r')
        D()
        result.append('D')
        R()
        result.append('R')
    elif color[2][1][2]==color[3][1][1] and color[4][1][2]==color[2][1][1]:
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
    elif color[0][0][1]==color[2][1][1] and color[5][1][0]==color[3][1][1]:
        l()
        result.append('l')
        r()
        result.append('r')
        D()
        result.append('D')
        D()
        result.append('D')
        R()
        result.append('R')
        L()
        result.append('L')
    elif color[0][0][1]==color[3][1][1] and color[5][1][0]==color[2][1][1]:
        l()
        result.append('l')
        D()
        result.append('D')
        L()
        result.append('L')
        d()
        result.append('d')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
    elif color[0][2][1]==color[2][1][1] and color[3][1][0]==color[3][1][1]:
        L()
        result.append('L')
        R()
        result.append('R')
        D()
        result.append('D')
        D()
        result.append('D')
        r()
```

```
        result.append('r')
        l()
        result.append('l')
    elif color[0][2][1]==color[3][1][1] and color[3][1][0]==color[2][1][1]:
        f()
        result.append('f')
        D()
        result.append('D')
        F()
        result.append('F')
        d()
        result.append('d')
        r()
        result.append('r')
        D()
        result.append('D')
        R()
        result.append('R')
    elif color[0][1][0]==color[2][1][1] and color[4][1][0]==color[3][1][1]:
        r()
        result.append('r')
        D()
        result.append('D')
        D()
        result.append('D')
        R()
        result.append('R')
    elif color[0][1][0]==color[3][1][1] and color[4][1][0]==color[2][1][1]:
        F()
        result.append('F')
        D()
        result.append('D')
        f()
        result.append('f')
    elif color[3][2][1]==color[2][1][1] and color[4][0][1]==color[3][1][1]:
        r()
        result.append('r')
        D()
        result.append('D')
        R()
        result.append('R')
    elif color[3][2][1]==color[3][1][1] and color[4][0][1]==color[2][1][1]:
        D()
        result.append('D')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
    elif color[5][0][1]==color[2][1][1] and color[4][2][1]==color[3][1][1]:
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
    elif color[5][0][1]==color[3][1][1] and color[4][2][1]==color[2][1][1]:
        F()
        result.append('F')
        D()
        result.append('D')
```

```
        D()
        result.append('D')
        f()
        result.append('f')

result.append('+')


# Top 4 corners

if not color[0][0][2]==color[0][1][1] or not color[1][0][0]==color[1][1][1] or not color[5][2][
0]==color[5][1][1] :
    if (color[0][0][2]==color[0][1][1] or color[0][0][2]==color[1][1][1] or color[0][0][2]==col
or[5][1][1]) and (color[1][0][0]==color[0][1][1] or color[1][0][0]==color[1][1][1] or color[1][
0][0]==color[5][1][1]) and (color[5][2][0]==color[0][1][1] or color[5][2][0]==color[1][1][1] or
 color[5][2][0]==color[5][1][1]):
        h=0
        for h in range(3):
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
        if color[4][2][0]==color[1][1][1]:
            h=0
            for h in range(3):
                l()
                result.append('l')
                d()
                result.append('d')
                L()
                result.append('L')
                D()
                result.append('D')
    elif color[0][0][0]==color[1][1][1]:
        d()
        result.append('d')
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
    elif color[5][0][0]==color[1][1][1]:
        D()
        result.append('D')
        B()
```

```
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
    elif (color[0][2][2]==color[0][1][1] or color[0][2][2]==color[1][1][1] or color[0][2][2]==c
olor[5][1][1]) and (color[1][2][0]==color[0][1][1] or color[1][2][0]==color[1][1][1] or color[1]
[2][0]==color[5][1][1]) and (color[3][0][0]==color[0][1][1] or color[3][0][0]==color[1][1][1] o
r color[3][0][0]==color[5][1][1]):
        h=0
        for h in range(3):
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
d()
result.append('d')
if color[4][2][0]==color[1][1][1]:
    h=0
    for h in range(3):
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
elif color[0][0][0]==color[1][1][1]:
    d()
    result.append('d')
    l()
    result.append('l')
    d()
    result.append('d')
    L()
    result.append('L')
    D()
    result.append('D')
    D()
    result.append('D')
    l()
    result.append('l')
    d()
    result.append('d')
    L()
```

```
            result.append('L')
            D()
            result.append('D')
        elif color[5][0][0]==color[1][1][1]:
            D()
            result.append('D')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
    elif (color[2][0][0]==color[0][1][1] or color[2][0][0]==color[1][1][1] or color[2][0][0]==c
olor[5][1][1]) and (color[1][0][2]==color[0][1][1] or color[1][0][2]==color[1][1][1] or color[1]
[0][2]==color[5][1][1]) and (color[5][2][2]==color[0][1][1] or color[5][2][2]==color[1][1][1] o
r color[5][2][2]==color[5][1][1]):
        h=0
        for h in range (3):
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
D()
result.append('D')
if color[4][2][0]==color[1][1][1]:
    h=0
    for h in range(3):
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
elif color[0][0][0]==color[1][1][1]:
    d()
    result.append('d')
    l()
    result.append('l')
    d()
    result.append('d')
    L()
    result.append('L')
    D()
    result.append('D')
```

```
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
    elif color[5][0][0]==color[1][1][1]:
        D()
        result.append('D')
        B()
        result.append('B')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
    elif (color[2][2][0]==color[0][1][1] or color[2][2][0]==color[1][1][1] or color[2][2][0]==c
olor[5][1][1]) and (color[1][2][2]==color[0][1][1] or color[1][2][2]==color[1][1][1] or color[1]
[2][2]==color[5][1][1]) and (color[3][0][2]==color[0][1][1] or color[3][0][2]==color[1][1][1] o
r color[3][0][2]==color[5][1][1]):
        h=0
        for h in range (3):
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
        D()
        result.append('D')
        D()
        result.append('D')
        if color[4][2][0]==color[1][1][1]:
            h=0
            for h in range(3):
                l()
                result.append('l')
                d()
                result.append('d')
                L()
                result.append('L')
                D()
                result.append('D')
        elif color[0][0][0]==color[1][1][1]:
            d()
```

```
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
            D()
            result.append('D')
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
        elif color[5][0][0]==color[1][1][1]:
            D()
            result.append('D')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
        elif (color[0][0][0]==color[0][1][1] or color[0][0][0]==color[1][1][1] or color[0][0][0]==c
olor[5][1][1]) and (color[5][0][0]==color[0][1][1] or color[5][0][0]==color[1][1][1] or color[5]
[0][0]==color[5][1][1]) and (color[4][2][0]==color[0][1][1] or color[4][2][0]==color[1][1][1] o
r color[4][2][0]==color[5][1][1]):
            if color[4][2][0]==color[1][1][1]:
                h=0
                for h in range(3):
                    l()
                    result.append('l')
                    d()
                    result.append('d')
                    L()
                    result.append('L')
                    D()
                    result.append('D')
            elif color[0][0][0]==color[1][1][1]:
                d()
                result.append('d')
                l()
                result.append('l')
                d()
                result.append('d')
```

```
        L()
        result.append('L')
        D()
        result.append('D')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
    elif color[5][0][0]==color[1][1][1]:
        D()
        result.append('D')
        B()
        result.append('B')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
    elif (color[0][2][0]==color[0][1][1] or color[0][2][0]==color[1][1][1] or color[0][2][0]==c
olor[5][1][1]) and (color[3][2][0]==color[0][1][1] or color[3][2][0]==color[1][1][1] or color[3]
[2][0]==color[5][1][1]) and (color[4][0][0]==color[0][1][1] or color[4][0][0]==color[1][1][1] o
r color[4][0][0]==color[5][1][1]):
        d()
        result.append('d')
        if color[4][2][0]==color[1][1][1]:
            h=0
            for h in range(3):
                l()
                result.append('l')
                d()
                result.append('d')
                L()
                result.append('L')
                D()
                result.append('D')
        elif color[0][0][0]==color[1][1][1]:
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
```

```
            result.append('D')
            D()
            result.append('D')
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
        elif color[5][0][0]==color[1][1][1]:
            D()
            result.append('D')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
        elif (color[2][0][2]==color[0][1][1] or color[2][0][2]==color[1][1][1] or color[2][0][2]==c
olor[5][1][1]) and (color[5][0][2]==color[0][1][1] or color[5][0][2]==color[1][1][1] or color[5]
[0][2]==color[5][1][1]) and (color[4][2][2]==color[0][1][1] or color[4][2][2]==color[1][1][1] o
r color[4][2][2]==color[5][1][1]):
            D()
            result.append('D')
            if color[4][2][0]==color[1][1][1]:
                h=0
                for h in range(3):
                    l()
                    result.append('l')
                    d()
                    result.append('d')
                    L()
                    result.append('L')
                    D()
                    result.append('D')
            elif color[0][0][0]==color[1][1][1]:
                d()
                result.append('d')
                l()
                result.append('l')
                d()
                result.append('d')
                L()
                result.append('L')
                D()
                result.append('D')
                D()
                result.append('D')
```

```
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
        elif color[5][0][0]==color[1][1][1]:
            D()
            result.append('D')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
    elif (color[2][2][2]==color[0][1][1] or color[2][2][2]==color[1][1][1] or color[2][2][2]==c
olor[5][1][1]) and (color[3][2][2]==color[0][1][1] or color[3][2][2]==color[1][1][1] or color[3]
[2][2]==color[5][1][1]) and (color[4][0][2]==color[0][1][1] or color[4][0][2]==color[1][1][1] o
r color[4][0][2]==color[5][1][1]):
        D()
        result.append('D')
        D()
        result.append('D')
        if color[4][2][0]==color[1][1][1]:
            h=0
            for h in range(3):
                l()
                result.append('l')
                d()
                result.append('d')
                L()
                result.append('L')
                D()
                result.append('D')
        elif color[0][0][0]==color[1][1][1]:
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
            D()
            result.append('D')
            l()
```

```
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
        elif color[5][0][0]==color[1][1][1]:
            D()
            result.append('D')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')

result.append('+')
if not color[0][2][2]==color[0][1][1] or not color[1][2][0]==color[1][1][1] or not color[3][0][
0]==color[3][1][1] :
    if (color[0][2][2]==color[0][1][1] or color[0][2][2]==color[1][1][1] or color[0][2][2]==col
or[3][1][1]) and (color[1][2][0]==color[0][1][1] or color[1][2][0]==color[1][1][1] or color[1][
2][0]==color[3][1][1]) and (color[3][0][0]==color[0][1][1] or color[3][0][0]==color[1][1][1] or
 color[3][0][0]==color[3][1][1]):
        h=0
        for h in range (3):
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
        if color[4][0][0]==color[1][1][1]:
            h=0
            for h in range (3):
                f()
                result.append('f')
                d()
                result.append('d')
                F()
                result.append('F')
                D()
                result.append('D')
        elif color[3][2][0]==color[1][1][1]:
            d()
            result.append('d')
            f()
```

```
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
        elif color[0][2][0]==color[1][1][1]:
            D()
            result.append('D')
            L()
            result.append('L')
            D()
            result.append('D')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
            l()
            result.append('l')
            d()
            result.append('d')
        elif (color[0][0][2]==color[0][1][1] or color[0][0][2]==color[1][1][1] or color[0][0][2]==c
olor[3][1][1]) and (color[1][0][0]==color[0][1][1] or color[1][0][0]==color[1][1][1] or color[1]
[0][0]==color[3][1][1]) and (color[5][2][0]==color[0][1][1] or color[5][2][0]==color[1][1][1] o
r color[5][2][0]==color==[3][1][1]):
            h=0
            for h in range(3):
                l()
                result.append('l')
                d()
                result.append('d')
                L()
                result.append('L')
                D()
                result.append('D')
            D()
            result.append('D')
            if color[4][0][0]==color[1][1][1]:
                h=0
                for h in range (3):
                    f()
                    result.append('f')
                    d()
                    result.append('d')
                    F()
```

```
            result.append('F')
            D()
            result.append('D')
    elif color[3][2][0]==color[1][1][1]:
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
    elif color[0][2][0]==color[1][1][1]:
        D()
        result.append('D')
        L()
        result.append('L')
        D()
        result.append('D')
                l()
                result.append('l')
                d()
                result.append('d')
                d()
                result.append('d')
                L()
                result.append('L')
                D()
                result.append('D')
                l()
                result.append('l')
                d()
                result.append('d')
        elif (color[2][0][0]==color[0][1][1] or color[2][0][0]==color[1][1][1] or color[2][0][0]==c
olor[3][1][1]) and (color[1][0][2]==color[0][1][1] or color[1][0][2]==color[1][1][1] or color[1]
[0][2]==color[3][1][1]) and (color[5][2][2]==color[0][1][1] or color[5][2][2]==color[1][1][1] o
r color[5][2][2]==color[3][1][1]):
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
                B()
                result.append('B')
                D()
                result.append('D')
            D()
            result.append('D')
            D()
```

```
result.append('D')
if color[4][0][0]==color[1][1][1]:
    h=0
    for h in range (3):
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
elif color[3][2][0]==color[1][1][1]:
    d()
    result.append('d')
    f()
    result.append('f')
    d()
    result.append('d')
    F()
    result.append('F')
    D()
    result.append('D')
    D()
    result.append('D')
    f()
    result.append('f')
    d()
    result.append('d')
    F()
    result.append('F')
        D()
        result.append('D')
    elif color[0][2][0]==color[1][1][1]:
        D()
        result.append('D')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
elif (color[2][2][0]==color[0][1][1] or color[2][2][0]==color[1][1][1] or color[2][2][0]==c
olor[3][1][1]) and (color[1][2][2]==color[0][1][1] or color[1][2][2]==color[1][1][1] or color[1]
[2][2]==color[3][1][1]) and (color[3][0][2]==color[0][1][1] or color[3][0][2]==color[1][1][1] o
r color[3][0][2]==color[3][1][1]):
    h=0
    for h in range (3):
        r()
        result.append('r')
```

```
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
    d()
    result.append('d')
    if color[4][0][0]==color[1][1][1]:
        h=0
        for h in range (3):
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
    elif color[3][2][0]==color[1][1][1]:
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
    elif color[0][2][0]==color[1][1][1]:
        D()
        result.append('D')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
```

```
    elif (color[0][2][0]==color[0][1][1] or color[0][2][0]==color[1][1][1] or color[0][2][0]==c
olor[3][1][1]) and (color[3][2][0]==color[0][1][1] or color[3][2][0]==color[1][1][1] or color[3]
[2][0]==color[3][1][1]) and (color[4][0][0]==color[0][1][1] or color[4][0][0]==color[1][1][1] o
r color[4][0][0]==color[3][1][1]):
        if color[4][0][0]==color[1][1][1]:
            h=0
            for h in range (3):
                f()
                result.append('f')
                d()
                result.append('d')
                F()
                result.append('F')
                D()
                result.append('D')
        elif color[3][2][0]==color[1][1][1]:
            d()
            result.append('d')
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
        elif color[0][2][0]==color[1][1][1]:
            D()
            result.append('D')
            L()
            result.append('L')
            D()
            result.append('D')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
            l()
            result.append('l')
            d()
            result.append('d')
    elif (color[0][0][0]==color[0][1][1] or color[0][0][0]==color[1][1][1] or color[0][0][0]==c
olor[3][1][1]) and (color[5][0][0]==color[0][1][1] or color[5][0][0]==color[1][1][1] or color[5]
[0][0]==color[3][1][1]) and (color[4][2][0]==color[0][1][1] or color[4][2][0]==color[1][1][1] o
r color[4][2][0]==color[3][1][1]):
        D()
```

```
result.append('D')
if color[4][0][0]==color[1][1][1]:
    h=0
    for h in range (3):
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
elif color[3][2][0]==color[1][1][1]:
    d()
    result.append('d')
    f()
    result.append('f')
    d()
    result.append('d')
    F()
    result.append('F')
    D()
    result.append('D')
    D()
    result.append('D')
    f()
    result.append('f')
    d()
    result.append('d')
    F()
    result.append('F')
        D()
        result.append('D')
    elif color[0][2][0]==color[1][1][1]:
        D()
        result.append('D')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
    elif (color[2][0][2]==color[0][1][1] or color[2][0][2]==color[1][1][1] or color[2][0][2]==c
olor[3][1][1]) and (color[5][0][2]==color[0][1][1] or color[5][0][2]==color[1][1][1] or color[5]
[0][2]==color[3][1][1]) and (color[4][2][2]==color[0][1][1] or color[4][2][2]==color[1][1][1] o
r color[4][2][2]==color[3][1][1]):
        D()
        result.append('D')
        D()
        result.append('D')
```

```
if color[4][0][0]==color[1][1][1]:
    h=0
    for h in range (3):
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
elif color[3][2][0]==color[1][1][1]:
    d()
    result.append('d')
    f()
    result.append('f')
    d()
    result.append('d')
    F()
    result.append('F')
    D()
    result.append('D')
    D()
    result.append('D')
    f()
    result.append('f')
    d()
    result.append('d')
    F()
    result.append('F')
    D()
        result.append('D')
    elif color[0][2][0]==color[1][1][1]:
        D()
        result.append('D')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
    elif (color[2][2][2]==color[0][1][1] or color[2][2][2]==color[1][1][1] or color[2][2][2]==c
olor[3][1][1]) and (color[3][2][2]==color[0][1][1] or color[3][2][2]==color[1][1][1] or color[3]
[2][2]==color[3][1][1]) and (color[4][0][2]==color[0][1][1] or color[4][0][2]==color[1][1][1] o
r color[4][0][2]==color[3][1][1]):
        d()
        result.append('d')
        if color[4][0][0]==color[1][1][1]:
            h=0
            for h in range (3):
```

```
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
    elif color[3][2][0]==color[1][1][1]:
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
    elif color[0][2][0]==color[1][1][1]:
        D()
        result.append('D')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
        l()
        result.append('l')
        d()
        result.append('d')

result.append('+')

if not color[2][0][0]==color[2][1][1] or not color[1][0][2]==color[1][1][1] or not color[5][2][
2]==color[5][1][1] :
    if (color[2][0][0]==color[2][1][1] or color[2][0][0]==color[1][1][1] or color[2][0][0]==col
or[5][1][1]) and (color[1][0][2]==color[2][1][1] or color[1][0][2]==color[1][1][1] or color[1][
0][2]==color[5][1][1]) and (color[5][2][2]==color[2][1][1] or color[5][2][2]==color[1][1][1] or
 color[5][2][2]==color[5][1][1]):
        h=0
        for h in range (3):
            b()
```

```
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
    if color[4][2][2]==color[1][1][1]:
        h=0
        for h in range (3):
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
    elif color[5][0][2]==color[1][1][1]:
        d()
        result.append('d')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
        D()
        result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
        elif color[2][0][2]==color[1][1][1]:
            D()
            result.append('D')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
    elif (color[2][2][0]==color[2][1][1] or color[2][2][0]==color[1][1][1] or color[2][2][0]==c
olor[5][1][1]) and (color[1][2][2]==color[2][1][1] or color[1][2][2]==color[1][1][1] or color[1]
```

```
[2][2]==color[5][1][1]) and (color[3][0][2]==color[2][1][1] or color[3][0][2]==color[1][1][1] o
r color[3][0][2]==color[5][1][1]):
        h=0
        for h in range (3):
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
        D()
        result.append('D')
        if color[4][2][2]==color[1][1][1]:
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
                B()
                result.append('B')
                D()
                result.append('D')
        elif color[5][0][2]==color[1][1][1]:
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
    elif color[2][0][2]==color[1][1][1]:
        D()
        result.append('D')
        R()
        result.append('R')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
```

```
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
    elif (color[0][0][2]==color[2][1][1] or color[0][0][2]==color[1][1][1] or color[0][0][2]==c
olor[5][1][1]) and (color[1][0][0]==color[2][1][1] or color[1][0][0]==color[1][1][1] or color[1]
[0][0]==color[5][1][1]) and (color[5][2][0]==color[2][1][1] or color[5][2][0]==color[1][1][1] o
r color[5][2][0]==color[5][1][1]):
        h=0
        for h in range(3):
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
        d()
        result.append('d')
        if color[4][2][2]==color[1][1][1]:
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
                B()
                result.append('B')
                D()
                result.append('D')
    elif color[5][0][2]==color[1][1][1]:
        d()
        result.append('d')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
    elif color[2][0][2]==color[1][1][1]:
        D()
        result.append('D')
        R()
        result.append('R')
        D()
        result.append('D')
        r()
        result.append('r')
```

```
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
    elif (color[0][2][2]==color[2][1][1] or color[0][2][2]==color[1][1][1] or color[0][2][2]==c
olor[5][1][1]) and (color[1][2][0]==color[2][1][1] or color[1][2][0]==color[1][1][1] or color[1]
[2][0]==color[5][1][1]) and (color[3][0][0]==color[2][1][1] or color[3][0][0]==color[1][1][1] o
r color[3][0][0]==color[5][1][1]):
        h=0
        for h in range(3):
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
        D()
        result.append('D')
        D()
        result.append('D')
        if color[4][2][2]==color[1][1][1]:
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
                B()
                result.append('B')
                D()
                result.append('D')
        elif color[5][0][2]==color[1][1][1]:
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
```

```
        elif color[2][0][2]==color[1][1][1]:
            D()
            result.append('D')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
    elif (color[2][0][2]==color[2][1][1] or color[2][0][2]==color[1][1][1] or color[2][0][2]==c
olor[5][1][1]) and (color[5][0][2]==color[2][1][1] or color[5][0][2]==color[1][1][1] or color[5]
[0][2]==color[5][1][1]) and (color[4][2][2]==color[2][1][1] or color[4][2][2]==color[1][1][1] o
r color[4][2][2]==color[5][1][1]):
        if color[4][2][2]==color[1][1][1]:
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
    elif color[5][0][2]==color[1][1][1]:
        d()
        result.append('d')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
    elif color[2][0][2]==color[1][1][1]:
        D()
        result.append('D')
        R()
        result.append('R')
```

```
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
    elif (color[2][2][2]==color[2][1][1] or color[2][2][2]==color[1][1][1] or color[2][2][2]==c
olor[5][1][1]) and (color[3][2][2]==color[2][1][1] or color[3][2][2]==color[1][1][1] or color[3]
[2][2]==color[5][1][1]) and (color[4][0][2]==color[2][1][1] or color[4][0][2]==color[1][1][1] o
r color[4][0][2]==color[5][1][1]):
        D()
        result.append('D')
        if color[4][2][2]==color[1][1][1]:
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
                B()
                result.append('B')
        D()
        result.append('D')
    elif color[5][0][2]==color[1][1][1]:
        d()
        result.append('d')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
        D()
        result.append('D')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
    elif color[2][0][2]==color[1][1][1]:
        D()
        result.append('D')
        R()
        result.append('R')
        D()
        result.append('D')
        r()
```

```
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
    elif (color[0][0][0]==color[2][1][1] or color[0][0][0]==color[1][1][1] or color[0][0][0]==c
olor[5][1][1]) and (color[5][0][0]==color[2][1][1] or color[5][0][0]==color[1][1][1] or color[5]
[0][0]==color[5][1][1]) and (color[4][2][0]==color[2][1][1] or color[4][2][0]==color[1][1][1] o
r color[4][2][0]==color[5][1][1]):
        d()
        result.append('d')
        if color[4][2][2]==color[1][1][1]:
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
                B()
                result.append('B')
                D()
                result.append('D')
        elif color[5][0][2]==color[1][1][1]:
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
    elif color[2][0][2]==color[1][1][1]:
        D()
        result.append('D')
        R()
        result.append('R')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
```

```
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
    elif (color[0][2][0]==color[2][1][1] or color[0][2][0]==color[1][1][1] or color[0][2][0]==c
olor[5][1][1]) and (color[3][2][0]==color[2][1][1] or color[3][2][0]==color[1][1][1] or color[3]
[2][0]==color[5][1][1]) and (color[4][0][0]==color[2][1][1] or color[4][0][0]==color[1][1][1] o
r color[4][0][0]==color[5][1][1]):
        D()
        result.append('D')
        D()
        result.append('D')
        if color[4][2][2]==color[1][1][1]:
            h=0
            for h in range (3):
                b()
                result.append('b')
                d()
                result.append('d')
                B()
                result.append('B')
                D()
                result.append('D')
        elif color[5][0][2]==color[1][1][1]:
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
            D()
            result.append('D')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            D()
            result.append('D')
    elif color[2][0][2]==color[1][1][1]:
        D()
        result.append('D')
        R()
        result.append('R')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
        d()
```

```
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')

result.append('+')

if not color[2][2][0]==color[2][1][1] or not color[1][2][2]==color[1][1][1] or not color[3][0][
2]==color[3][1][1] :
    if (color[2][2][0]==color[2][1][1] or color[2][2][0]==color[1][1][1] or color[2][2][0]==col
or[3][1][1]) and (color[1][2][2]==color[2][1][1] or color[1][2][2]==color[1][1][1] or color[1][
2][2]==color[3][1][1]) and (color[3][0][2]==color[2][1][1] or color[3][0][2]==color[1][1][1] or
 color[3][0][2]==color[3][1][1]):
        h=0
        for h in range (3):
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
        if color[4][0][2]==color[1][1][1]:
            h=0
            for h in range (3):
                r()
                result.append('r')
                d()
                result.append('d')
                R()
                result.append('R')
                D()
                result.append('D')
    elif color[2][2][2]==color[1][1][1]:
        d()
        result.append('d')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
    elif color[3][2][2]==color[1][1][1]:
        D()
```

```
            result.append('D')
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
    elif (color[2][0][0]==color[2][1][1] or color[2][0][0]==color[1][1][1] or color[2][0][0]==c
olor[3][1][1]) and (color[1][0][2]==color[2][1][1] or color[1][0][2]==color[1][1][1] or color[1]
[0][2]==color[3][1][1]) and (color[5][2][2]==color[2][1][1] or color[5][2][2]==color[1][1][1] o
r color[5][2][2]==color[3][1][1]):
        h=0
        for h in range (3):
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
        D()
        result.append('D')
    d()
    result.append('d')
    if color[4][0][2]==color[1][1][1]:
        h=0
        for h in range (3):
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
    elif color[2][2][2]==color[1][1][1]:
        d()
        result.append('d')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
```

```
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
        elif color[3][2][2]==color[1][1][1]:
            D()
            result.append('D')
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
    elif (color[0][0][2]==color[2][1][1] or color[0][0][2]==color[1][1][1] or color[0][0][2]==c
olor[3][1][1]) and (color[1][0][0]==color[2][1][1] or color[1][0][0]==color[1][1][1] or color[1]
[0][0]==color[3][1][1]) and (color[5][2][0]==color[2][1][1] or color[5][2][0]==color[1][1][1] o
r color[5][2][0]==color==[3][1][1]):
        h=0
        for h in range(3):
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            D()
            result.append('D')
        D()
        result.append('D')
        D()
        result.append('D')
        if color[4][0][2]==color[1][1][1]:
            h=0
            for h in range (3):
                r()
                result.append('r')
                d()
                result.append('d')
                R()
                result.append('R')
                D()
                result.append('D')
        elif color[2][2][2]==color[1][1][1]:
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
```

```
            R()
            result.append('R')
            D()
            result.append('D')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
        elif color[3][2][2]==color[1][1][1]:
            D()
            result.append('D')
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
                f()
                result.append('f')
                d()
                result.append('d')
    elif (color[0][2][2]==color[2][1][1] or color[0][2][2]==color[1][1][1] or color[0][2][2]==c
olor[3][1][1]) and (color[1][2][0]==color[2][1][1] or color[1][2][0]==color[1][1][1] or color[1]
[2][0]==color[3][1][1]) and (color[3][0][0]==color[2][1][1] or color[3][0][0]==color[1][1][1] o
r color[3][0][0]==color[3][1][1]):
            h=0
            for h in range (3):
                f()
                result.append('f')
                d()
                result.append('d')
                F()
                result.append('F')
                D()
                result.append('D')
            D()
            result.append('D')
            if color[4][0][2]==color[1][1][1]:
                h=0
                for h in range (3):
                    r()
                    result.append('r')
                    d()
                    result.append('d')
                    R()
                    result.append('R')
                    D()
                    result.append('D')
```

```
elif color[2][2][2]==color[1][1][1]:
    d()
    result.append('d')
    r()
    result.append('r')
    d()
    result.append('d')
    R()
    result.append('R')
    D()
    result.append('D')
    D()
    result.append('D')
    r()
    result.append('r')
    d()
    result.append('d')
    R()
    result.append('R')
    D()
    result.append('D')
elif color[3][2][2]==color[1][1][1]:
    D()
    result.append('D')
    F()
    result.append('F')
    D()
    result.append('D')
    f()
    result.append('f')
    d()
    result.append('d')
    d()
    result.append('d')
    F()
    result.append('F')
    D()
    result.append('D')
    f()
    result.append('f')
    d()
    result.append('d')
elif (color[2][2][2]==color[2][1][1] or color[2][2][2]==color[1][1][1] or color[2][2][2]==c
olor[3][1][1]) and (color[3][2][2]==color[2][1][1] or color[3][2][2]==color[1][1][1] or color[3]
[2][2]==color[3][1][1]) and (color[4][0][2]==color[2][1][1] or color[4][0][2]==color[1][1][1] o
r color[4][0][2]==color[3][1][1]):
    if color[4][0][2]==color[1][1][1]:
        h=0
        for h in range (3):
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
    elif color[2][2][2]==color[1][1][1]:
        d()
        result.append('d')
        r()
        result.append('r')
```

```
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
    elif color[3][2][2]==color[1][1][1]:
        D()
        result.append('D')
        F()
        result.append('F')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
    elif (color[2][0][2]==color[2][1][1] or color[2][0][2]==color[1][1][1] or color[2][0][2]==c
olor[3][1][1]) and (color[5][0][2]==color[2][1][1] or color[5][0][2]==color[1][1][1] or color[5]
[0][2]==color[3][1][1]) and (color[4][2][2]==color[2][1][1] or color[4][2][2]==color[1][1][1] o
r color[4][2][2]==color[3][1][1]):
        d()
        result.append('d')
        if color[4][0][2]==color[1][1][1]:
            h=0
            for h in range (3):
                r()
                result.append('r')
                d()
                result.append('d')
                R()
                result.append('R')
                D()
                result.append('D')
        elif color[2][2][2]==color[1][1][1]:
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            R()
```

```
            result.append('R')
            D()
            result.append('D')
            D()
            result.append('D')
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
        elif color[3][2][2]==color[1][1][1]:
            D()
            result.append('D')
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
            D()
            result.append('D')
            f()
            result.append('f')
            d()
            result.append('d')
        elif (color[0][0][0]==color[2][1][1] or color[0][0][0]==color[1][1][1] or color[0][0][0]==c
olor[3][1][1]) and (color[5][0][0]==color[2][1][1] or color[5][0][0]==color[1][1][1] or color[5]
[0][0]==color[3][1][1]) and (color[4][2][0]==color[2][1][1] or color[4][2][0]==color[1][1][1] o
r color[4][2][0]==color[3][1][1]):
            D()
            result.append('D')
            D()
            result.append('D')
            if color[4][0][2]==color[1][1][1]:
                h=0
                for h in range (3):
                    r()
                    result.append('r')
                    d()
                    result.append('d')
                    R()
                    result.append('R')
                    D()
                    result.append('D')
            elif color[2][2][2]==color[1][1][1]:
                d()
                result.append('d')
                r()
                result.append('r')
                d()
                result.append('d')
                R()
                result.append('R')
```

```
        D()
        result.append('D')
        D()
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
    elif color[3][2][2]==color[1][1][1]:
        D()
        result.append('D')
        F()
        result.append('F')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
    elif (color[0][2][0]==color[2][1][1] or color[0][2][0]==color[1][1][1] or color[0][2][0]==c
olor[3][1][1]) and (color[3][2][0]==color[2][1][1] or color[3][2][0]==color[1][1][1] or color[3]
[2][0]==color[3][1][1]) and (color[4][0][0]==color[2][1][1] or color[4][0][0]==color[1][1][1] o
r color[4][0][0]==color[3][1][1]):
        D()
        result.append('D')
        if color[4][0][2]==color[1][1][1]:
            h=0
            for h in range (3):
                r()
                result.append('r')
                d()
                result.append('d')
                R()
                result.append('R')
                D()
                result.append('D')
        elif color[2][2][2]==color[1][1][1]:
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            D()
            result.append('D')
            D()
```

```
        result.append('D')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
    elif color[3][2][2]==color[1][1][1]:
        D()
        result.append('D')
        F()
        result.append('F')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
        f()
        result.append('f')
        d()
        result.append('d')
result.append('+')


# Bottom Cross

if not color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4][2][1] and ((not
 color[4][0][1]==color[4][1][1] and not color[4][1][0]==color[4][1][1] and not color[4][1][2]==
color[4][1][1] and not color[4][2][1]==color[4][1][1]) or (color[4][0][1]==color[4][1][1] and n
ot color[4][1][0]==color[4][1][1] and not color[4][1][2]==color[4][1][1] and not color[4][2][1]
==color[4][1][1]) or (color[4][1][0]==color[4][1][1] and not color[4][0][1]==color[4][1][1] and
 not color[4][1][2]==color[4][1][1] and not color[4][2][1]==color[4][1][1]) or (color[4][1][2]=
=color[4][1][1] and not color[4][1][0]==color[4][1][1] and not color[4][0][1]==color[4][1][1] a
nd not color[4][2][1]==color[4][1][1]) or (color[4][2][1]==color[4][1][1] and not color[4][1][0]
==color[4][1][1] and not color[4][1][2]==color[4][1][1] and not color[4][0][1]==color[4][1][1]))
 :
    f()
    result.append('f')
    r()
    result.append('r')
    d()
    result.append('d')
    R()
    result.append('R')
    D()
    result.append('D')
    F()
    result.append('F')
if not color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4][2][1] and (((co
lor[4][0][1]==color[4][1][1]==color[4][1][2]==color[4][2][1] and not color[4][1][0]==color[4][1]
[1]) or (color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2] and not color[4][2][1]=
=color[4][1][1]) or (color[4][0][1]==color[4][1][1]==color[4][1][2] and not color[4][1][0]==col
```

```
or[4][1][1] and not color[4][2][1]==color[4][1][1])) or ((color[4][0][1]==color[4][1][0]==color
[4][1][1]==color[4][2][1] and not color[4][1][2]==color[4][1][1]) or (color[4][1][0]==color[4][
1][1]==color[4][1][2]==color[4][2][1] and not color[4][0][1]==color[4][1][1]) or (color[4][1][0]
==color[4][1][1]==color[4][2][1] and not color[4][0][1]==color[4][1][1] and not color[4][1][2]=
=color[4][1][1])) or (color[4][0][1]==color[4][1][0]==color[4][1][1] and not color[4][1][2]==co
lor[4][1][1] and not color[4][2][1]==color[4][1][1]) or (color[4][1][1]==color[4][1][2]==color[
4][2][1] and not color[4][0][1]==color[4][1][1] and not color[4][1][0]==color[4][1][1])) :
    if (color[4][0][1]==color[4][1][1]==color[4][1][2]==color[4][2][1] and not color[4][1][0]==
color[4][1][1]) or (color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2] and not colo
r[4][2][1]==color[4][1][1]) or (color[4][0][1]==color[4][1][1]==color[4][1][2] and not color[4]
[1][0]==color[4][1][1] and not color[4][2][1]==color[4][1][1]) :
        f()
        result.append('f')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
        F()
        result.append('F')
    elif (color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][2][1] and not color[4][1][2]
==color[4][1][1]) or (color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4][2][1] and not co
lor[4][0][1]==color[4][1][1]) or (color[4][1][0]==color[4][1][1]==color[4][2][1] and not color[
4][0][1]==color[4][1][1] and not color[4][1][2]==color[4][1][1]) :
        b()
        result.append('b')
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        D()
        result.append('D')
        B()
        result.append('B')
    elif color[4][0][1]==color[4][1][0]==color[4][1][1] and not color[4][1][2]==color[4][1][1]
and not color[4][2][1]==color[4][1][1] :
        l()
        result.append('l')
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        D()
        result.append('D')
        L()
        result.append('L')
    elif color[4][1][1]==color[4][1][2]==color[4][2][1] and not color[4][0][1]==color[4][1][1]
and not color[4][1][0]==color[4][1][1] :
        r()
        result.append('r')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
```

```
        result.append('B')
        D()
        result.append('D')
        R()
        result.append('R')
if not color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4][2][1] and (colo
r[4][0][1]==color[4][1][1]==color[4][2][1] or color[4][1][0]==color[4][1][1]==color[4][1][2]) :
    if color[4][0][1]==color[4][1][1]==color[4][2][1] :
        r()
        result.append('r')
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        D()
        result.append('D')
        R()
        result.append('R')
    elif color[4][1][0]==color[4][1][1]==color[4][1][2] :
        f()
        result.append('f')
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        D()
        result.append('D')
        F()
        result.append('F')

result.append('+')

# OLL

if (color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4][2][0]
==color[4][1][1] and not color[4][2][2]==color[4][1][1]) or (color[4][0][2]==color[4][1][1] and
 not color[4][0][0]==color[4][1][1] and not color[4][2][0]==color[4][1][1] and not color[4][2][
2]==color[4][1][1]) or (color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] a
nd not color[4][0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1]) or (color[4][2][2]
==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4][2][0]==color[4][1][1]
and not color[4][0][0]==color[4][1][1]) :
    if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4][2]
[0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        d()
        result.append('d')
        r()
        result.append('r')
        d()
        result.append('d')
        d()
        result.append('d')
        R()
```

```
        result.append('R')
    elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not color[4]
[2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        d()
        result.append('d')
        b()
        result.append('b')
        d()
        result.append('d')
        d()
        result.append('d')
        B()
        result.append('B')
    elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
    elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        d()
        result.append('d')
        l()
        result.append('l')
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
    if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4][2]
[0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
        r()
        result.append('r')
        d()
        result.append('d')
        R()
```

```
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not color[4]
[2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
        elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            d()
            result.append('d')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
        elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
```

```
        result.append('d')
        L()
        result.append('L')
elif not color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
    if not color[3][2][0]==color[4][1][1] and not color[3][2][1]==color[4][1][1] and not color[
3][2][2]==color[4][1][1] :
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        d()
        result.append('d')
        r()
        result.append('r')
        d()
        result.append('d')
        d()
        result.append('d')
        R()
        result.append('R')
    elif not color[0][0][0]==color[4][1][1] and not color[0][1][0]==color[4][1][1] and not colo
r[0][2][0]==color[4][1][1] :
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
    elif not color[2][0][2]==color[4][1][1] and not color[2][1][2]==color[4][1][1] and not colo
r[2][2][2]==color[4][1][1] :
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        d()
        result.append('d')
        b()
        result.append('b')
        d()
        result.append('d')
        d()
        result.append('d')
        B()
        result.append('B')
    elif not color[5][0][0]==color[4][1][1] and not color[5][0][1]==color[4][1][1] and not colo
r[5][0][2]==color[4][1][1] :
        l()
```

```
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
        if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4][2]
    [0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not color[4]
    [2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
        elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
    [0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            d()
            result.append('d')
            f()
```

```
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
        elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
    [2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
        if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4][2]
    [0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not color[4]
    [2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
```

```
    elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
    elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        d()
        result.append('d')
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
elif (color[4][0][0]==color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4][
2][0]==color[4][2][1] and not color[4][0][2]==color[4][1][1] and not color[4][2][2]==color[4][1]
[1]) or (color[4][0][0]==color[4][0][1]==color[4][0][2]==color[4][1][0]==color[4][1][1]==color[
4][1][2]==color[4][2][1] and not color[4][2][0]==color[4][1][1] and not color[4][2][2]==color[4]
[1][1]) or (color[4][0][1]==color[4][0][2]==color[4][1][0]==color[4][1][1]==color[4][1][2]==col
or[4][2][1]==color[4][2][2] and not color[4][0][0]==color[4][1][1] and not color[4][2][0]==colo
r[4][1][1]) or (color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4][2][0]=
=color[4][2][1]==color[4][2][2] and not color[4][0][0]==color[4][1][1] and not color[4][0][2]==
color[4][1][1]) or (color[4][0][0]==color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1]
[2]==color[4][2][1]==color[4][2][2] and not color[4][0][2]==color[4][1][1] and not color[4][2][
0]==color[4][1][1]) or (color[4][0][1]==color[4][0][2]==color[4][1][0]==color[4][1][1]==color[4]
[1][2]==color[4][2][0]==color[4][2][1] and not color[4][0][0]==color[4][1][1] and not color[4][
2][2]==color[4][1][1]) :
    if color[4][0][0]==color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4]
[2][0]==color[4][2][1] and not color[4][0][2]==color[4][1][1] and not color[4][2][2]==color[4][
1][1] :
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        d()
        result.append('d')
        l()
        result.append('l')
```

```
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
    elif color[4][0][0]==color[4][0][1]==color[4][0][2]==color[4][1][0]==color[4][1][1]==color[
4][1][2]==color[4][2][1] and not color[4][2][0]==color[4][1][1] and not color[4][2][2]==color[4]
[1][1] :
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
    elif color[4][0][1]==color[4][0][2]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[
4][2][1]==color[4][2][2] and not color[4][0][0]==color[4][1][1] and not color[4][2][0]==color[4]
[1][1] :
        r()
        result.append('r')
        d()
        result.append('d')
        R()
        result.append('R')
        d()
        result.append('d')
        r()
        result.append('r')
        d()
        result.append('d')
        d()
        result.append('d')
        R()
        result.append('R')
    elif color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[4][2][0]==color[
4][2][1]==color[4][2][2] and not color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4]
[1][1] :
        b()
        result.append('b')
        d()
        result.append('d')
        B()
        result.append('B')
        d()
        result.append('d')
        b()
        result.append('b')
        d()
        result.append('d')
        d()
        result.append('d')
        B()
```

```
        result.append('B')
    elif color[4][0][0]==color[4][0][1]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[
4][2][1]==color[4][2][2] and not color[4][0][2]==color[4][1][1] and not color[4][2][0]==color[4]
[1][1] :
        f()
        result.append('f')
        d()
        result.append('d')
        F()
        result.append('F')
        d()
        result.append('d')
        f()
        result.append('f')
        d()
        result.append('d')
        d()
        result.append('d')
        F()
        result.append('F')
    elif color[4][0][1]==color[4][0][2]==color[4][1][0]==color[4][1][1]==color[4][1][2]==color[
4][2][0]==color[4][2][1] and not color[4][0][0]==color[4][1][1] and not color[4][2][2]==color[4]
[1][1] :
        l()
        result.append('l')
        d()
        result.append('d')
        L()
        result.append('L')
        d()
        result.append('d')
        l()
        result.append('l')
        d()
        result.append('d')
        d()
        result.append('d')
        L()
        result.append('L')
    if not color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[
4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
        if not color[3][2][0]==color[4][1][1] and not color[3][2][1]==color[4][1][1] and not co
lor[3][2][2]==color[4][1][1] :
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif not color[0][0][0]==color[4][1][1] and not color[0][1][0]==color[4][1][1] and not
color[0][2][0]==color[4][1][1] :
            f()
```

```
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            d()
            result.append('d')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
        elif not color[2][0][2]==color[4][1][1] and not color[2][1][2]==color[4][1][1] and not
color[2][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
        elif not color[2][2][2]==color[4][1][1] and not color[2][1][2]==color[4][1][1] and not
color[2][0][2]==color[4][1][1] :
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
        if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[
4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            d()
            result.append('d')
            r()
```

```
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
        elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            d()
            result.append('d')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
        elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
```

```
        if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[
4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
        elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            d()
            result.append('d')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
        elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
```

```
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
    elif (color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4]
[2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1]) or (color[4][0][2]==color[4][1][
1] and not color[4][0][0]==color[4][1][1] and not color[4][2][0]==color[4][1][1] and not color[
4][2][2]==color[4][1][1]) or (color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1]
[1] and not color[4][0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1]) or (color[4]
[2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[4][2][0]==color[4][
1][1] and not color[4][0][0]==color[4][1][1]) :
        if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[
4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
            result.append('B')
        elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            d()
```

```
            result.append('d')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
        elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')
        if color[4][0][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not color[
4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            r()
            result.append('r')
            d()
            result.append('d')
            R()
            result.append('R')
            d()
            result.append('d')
            r()
            result.append('r')
            d()
            result.append('d')
            d()
            result.append('d')
            R()
            result.append('R')
        elif color[4][0][2]==color[4][1][1] and not color[4][0][0]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            b()
            result.append('b')
            d()
            result.append('d')
            B()
            result.append('B')
            d()
            result.append('d')
            b()
            result.append('b')
            d()
            result.append('d')
            d()
            result.append('d')
            B()
```

```
            result.append('B')
        elif color[4][2][0]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][0][0]==color[4][1][1] and not color[4][2][2]==color[4][1][1] :
            f()
            result.append('f')
            d()
            result.append('d')
            F()
            result.append('F')
            d()
            result.append('d')
            f()
            result.append('f')
            d()
            result.append('d')
            d()
            result.append('d')
            F()
            result.append('F')
        elif color[4][2][2]==color[4][1][1] and not color[4][0][2]==color[4][1][1] and not colo
r[4][2][0]==color[4][1][1] and not color[4][0][0]==color[4][1][1] :
            l()
            result.append('l')
            d()
            result.append('d')
            L()
            result.append('L')
            d()
            result.append('d')
            l()
            result.append('l')
            d()
            result.append('d')
            d()
            result.append('d')
            L()
            result.append('L')

result.append('+')


# PLL

if not color[3][2][0]==color[3][2][2] and not color[0][0][0]==color[0][2][0] and not color[2][0]
[2]==color[2][2][2] and not color[5][0][0]==color[5][0][2] :
    r()
    result.append('r')
    B()
    result.append('B')
    r()
    result.append('r')
    F()
    result.append('F')
    F()
    result.append('F')
    R()
    result.append('R')
    b()
    result.append('b')
    r()
    result.append('r')
    F()
```

```
    result.append('F')
    F()
    result.append('F')
    R()
    result.append('R')
    R()
    result.append('R')
if color[3][2][0]==color[3][2][2] and (not color[0][0][0]==color[0][2][0] or not color[2][0][2]
==color[2][2][2] or not color[5][0][0]==color[5][0][2]) :
    if color[3][2][0]==color[3][2][2]==color[3][1][1] :
        r()
        result.append('r')
        B()
        result.append('B')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
        result.append('R')
        b()
        result.append('b')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[3][2][0]==color[3][2][2]==color[0][1][1] :
        d()
        result.append('d')
        f()
        result.append('f')
        R()
        result.append('R')
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        r()
        result.append('r')
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        F()
        result.append('F')
    elif color[3][2][0]==color[3][2][2]==color[2][1][1] :
```

```
        D()
        result.append('D')
        b()
        result.append('b')
        L()
        result.append('L')
        b()
        result.append('b')
        R()
        result.append('R')
        R()
        result.append('R')
        B()
        result.append('B')
        l()
        result.append('l')
        b()
        result.append('b')
        R()
        result.append('R')
        R()
        result.append('R')
        B()
        result.append('B')
        B()
        result.append('B')
    elif color[3][2][0]==color[3][2][2]==color[5][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        l()
        result.append('l')
        F()
        result.append('F')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        f()
        result.append('f')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        L()
        result.append('L')
elif color[0][0][0]==color[0][2][0] and (not color[3][2][0]==color[3][2][2] or not color[2][0][
2]==color[2][2][2] or not color[5][0][0]==color[5][0][2]) :
    if color[0][0][0]==color[0][2][0]==color[0][1][1] :
        f()
        result.append('f')
        R()
        result.append('R')
```

```
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        r()
        result.append('r')
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        F()
        result.append('F')
    elif color[0][0][0]==color[0][2][0]==color[5][1][1] :
        d()
        result.append('d')
        l()
        result.append('l')
        F()
        result.append('F')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        f()
        result.append('f')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[0][0][0]==color[0][2][0]==color[3][1][1] :
        D()
        result.append('D')
        r()
        result.append('r')
        B()
        result.append('B')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
        result.append('R')
```

```
        b()
        result.append('b')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[0][0][0]==color[0][2][0]==color[2][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        b()
        result.append('b')
        L()
        result.append('L')
        b()
        result.append('b')
        R()
        result.append('R')
        R()
        result.append('R')
        B()
        result.append('B')
        l()
        result.append('l')
        b()
        result.append('b')
        R()
        result.append('R')
        B()
        result.append('B')
        B()
        result.append('B')
elif color[2][0][2]==color[2][2][2] and (not color[0][0][0]==color[0][2][0] or not color[3][2][
0]==color[3][2][2] or not color[5][0][0]==color[5][0][2]) :
    if color[2][0][2]==color[2][2][2]==color[2][1][1] :
        b()
        result.append('b')
        L()
        result.append('L')
        b()
        result.append('b')
        R()
        result.append('R')
        R()
        result.append('R')
        B()
        result.append('B')
        l()
        result.append('l')
        b()
        result.append('b')
        R()
        result.append('R')
```

```
        R()
        result.append('R')
        B()
        result.append('B')
        B()
        result.append('B')
    elif color[2][0][2]==color[2][2][2]==color[3][1][1] :
        d()
        result.append('d')
        r()
        result.append('r')
        B()
        result.append('B')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
        result.append('R')
        b()
        result.append('b')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[2][0][2]==color[2][2][2]==color[5][1][1] :
        D()
        result.append('D')
        l()
        result.append('l')
        F()
        result.append('F')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        f()
        result.append('f')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[2][0][2]==color[2][2][2]==color[0][1][1] :
        D()
```

```
        result.append('D')
        D()
        result.append('D')
        f()
        result.append('f')
        R()
        result.append('R')
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        r()
        result.append('r')
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        F()
        result.append('F')
elif color[5][0][0]==color[5][0][2] and (not color[0][0][0]==color[0][2][0] or not color[2][0][
2]==color[2][2][2] or not color[3][2][0]==color[3][2][2]) :
    if color[5][0][0]==color[5][0][2]==color[5][1][1] :
        l()
        result.append('l')
        F()
        result.append('F')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        f()
        result.append('f')
        l()
        result.append('l')
        B()
        result.append('B')
        B()
        result.append('B')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[5][0][0]==color[5][0][2]==color[2][1][1] :
        d()
        result.append('d')
        b()
        result.append('b')
        L()
        result.append('L')
        b()
```

```
        result.append('b')
        R()
        result.append('R')
        R()
        result.append('R')
        B()
        result.append('B')
        l()
        result.append('l')
        b()
        result.append('b')
        R()
        result.append('R')
        R()
        result.append('R')
        B()
        result.append('B')
        B()
        result.append('B')
    elif color[5][0][0]==color[5][0][2]==color[0][1][1] :
        D()
        result.append('D')
        f()
        result.append('f')
        R()
        result.append('R')
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        r()
        result.append('r')
        f()
        result.append('f')
        L()
        result.append('L')
        L()
        result.append('L')
        F()
        result.append('F')
        F()
        result.append('F')
    elif color[5][0][0]==color[5][0][2]==color[3][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        r()
        result.append('r')
        B()
        result.append('B')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
```

```
        result.append('R')
        b()
        result.append('b')
        r()
        result.append('r')
        F()
        result.append('F')
        F()
        result.append('F')
        R()
        result.append('R')
        R()
        result.append('R')

if color[3][2][0]==color[3][2][2]==color[2][1][1]:
    D()
    result.append('D')
elif color[3][2][0]==color[3][2][2]==color[0][1][1]:
    d()
    result.append('d')
elif color[3][2][0]==color[3][2][2]==color[5][1][1]:
    D()
    result.append('D')
    D()
    result.append('D')

if (color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[0][0][0]==color[0][1][0]==colo
r[0][2][0] and not color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[5][0][0]==color
[5][0][1]==color[5][0][2]) or (color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][
2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==color[2][1][2]==color[2][2][2] an
d not color[5][0][0]==color[5][0][1]==color[5][0][2]) or (color[2][0][2]==color[2][1][2]==color
[2][2][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][2][0]==color[
3][2][1]==color[3][2][2] and not color[5][0][0]==color[5][0][1]==color[5][0][2]) or (color[5][0]
[0]==color[5][0][1]==color[5][0][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and
not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==color[2][1][2]==colo
r[2][2][2]) :
    if color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[0][0][0]==color[0][1][0]==c
olor[0][2][0] and not color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[5][0][0]==co
lor[5][0][1]==color[5][0][2] :
        if color[3][2][0]==color[3][2][1]==color[3][2][2]==color[3][1][1] :
            B()
            result.append('B')
            B()
            result.append('B')
            if color[5][1][1]==color[0][1][0] :
                d()
                result.append('d')
                temp = 'd'
            else :
                D()
                result.append('D')
                temp = 'D'
            L()
            result.append('L')
            r()
            result.append('r')
            B()
            result.append('B')
            B()
            result.append('B')
            l()
            result.append('l')
```

```
        R()
        result.append('R')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        B()
        result.append('B')
        B()
        result.append('B')
    elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[0][1][1] :
        d()
        result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')
        if color[2][1][1]==color[3][2][1] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        B()
        result.append('B')
        f()
        result.append('f')
        R()
        result.append('R')
        R()
        result.append('R')
        b()
        result.append('b')
        F()
        result.append('F')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[2][1][1] :
        D()
        result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
        if color[3][2][1]==color[0][1][1] :
            d()
            result.append('d')
            temp = 'd'
        else :
            D()
```

```
            result.append('D')
            temp = 'D'
        b()
        result.append('b')
        F()
        result.append('F')
        L()
        result.append('L')
        L()
        result.append('L')
        B()
        result.append('B')
        f()
        result.append('f')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[5][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        l()
        result.append('l')
        R()
        result.append('R')
        F()
        result.append('F')
        F()
        result.append('F')
        L()
        result.append('L')
        r()
        result.append('r')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        F()
        result.append('F')
        F()
```

```
            result.append('F')
    elif color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[5][0][0]==color[5][0][1]==color[5][0][2] :
        if color[0][0][0]==color[0][1][0]==color[0][2][0]==color[3][1][1] :
            D()
            result.append('D')
            B()
            result.append('B')
            B()
            result.append('B')
            if color[5][1][1]==color[0][1][0] :
                d()
                result.append('d')
                temp = 'd'
            else :
                D()
                result.append('D')
                temp = 'D'
            L()
            result.append('L')
            r()
            result.append('r')
            B()
            result.append('B')
            B()
            result.append('B')
            l()
            result.append('l')
            R()
            result.append('R')
            if temp == 'd' :
                d()
                result.append('d')
            else :
                D()
                result.append('D')
            B()
            result.append('B')
            B()
            result.append('B')
        elif color[0][0][0]==color[0][1][0]==color[0][2][0]==color[0][1][1] :
            R()
            result.append('R')
            R()
            result.append('R')
            if color[2][1][1]==color[3][2][1] :
                D()
                result.append('D')
                temp = 'D'
            else :
                d()
                result.append('d')
                temp = 'd'
            B()
            result.append('B')
            f()
            result.append('f')
            R()
            result.append('R')
            R()
            result.append('R')
```

```
        b()
        result.append('b')
        F()
        result.append('F')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[0][0][0]==color[0][1][0]==color[0][2][0]==color[2][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
        if color[3][2][1]==color[0][1][1] :
            d()
            result.append('d')
            temp = 'd'
        else :
            D()
            result.append('D')
            temp = 'D'
```

```
        b()
        result.append('b')
        F()
        result.append('F')
        L()
        result.append('L')
        L()
        result.append('L')
        B()
        result.append('B')
        f()
        result.append('f')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[0][0][0]==color[0][1][0]==color[0][2][0]==color[5][1][1] :
        d()
        result.append('d')
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
```

```
                result.append('D')
                temp = 'D'
            else :
                d()
                result.append('d')
                temp = 'd'
            l()
            result.append('l')
            R()
            result.append('R')
            F()
            result.append('F')
            F()
            result.append('F')
            L()
            result.append('L')
            r()
            result.append('r')
            if temp == 'D' :
                D()
                result.append('D')
            else :
                d()
                result.append('d')
            F()
            result.append('F')
            F()
            result.append('F')
    elif color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[5][0][0]==color[5][0][1]==color[5][0][2] :
        if color[2][0][2]==color[2][1][2]==color[2][2][2]==color[3][1][1] :
            d()
            result.append('d')
            B()
            result.append('B')
            B()
            result.append('B')
            if color[5][1][1]==color[0][1][0] :
                d()
                result.append('d')
                temp = 'd'
            else :
                D()
                result.append('D')
                temp = 'D'
            L()
            result.append('L')
            r()
            result.append('r')
            B()
            result.append('B')
            B()
            result.append('B')
            l()
            result.append('l')
            R()
            result.append('R')
            if temp == 'd' :
                d()
                result.append('d')
            else :
```

```
            D()
            result.append('D')
        B()
        result.append('B')
        B()
        result.append('B')
    elif color[2][0][2]==color[2][1][2]==color[2][2][2]==color[0][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        R()
        result.append('R')
        R()
        result.append('R')
        if color[2][1][1]==color[3][2][1] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        B()
        result.append('B')
        f()
        result.append('f')
        R()
        result.append('R')
        R()
        result.append('R')
        b()
        result.append('b')
        F()
        result.append('F')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[2][0][2]==color[2][1][2]==color[2][2][2]==color[2][1][1] :
        L()
        result.append('L')
        L()
        result.append('L')
        if color[3][2][1]==color[0][1][1] :
            d()
            result.append('d')
            temp = 'd'
        else :
            D()
            result.append('D')
            temp = 'D'
        b()
        result.append('b')
        F()
        result.append('F')
```

```
        L()
        result.append('L')
        L()
        result.append('L')
        B()
        result.append('B')
        f()
        result.append('f')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[2][0][2]==color[2][1][2]==color[2][2][2]==color[5][1][1] :
        D()
        result.append('D')
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        l()
        result.append('l')
        R()
        result.append('R')
        F()
        result.append('F')
        F()
        result.append('F')
        L()
        result.append('L')
        r()
        result.append('r')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        F()
        result.append('F')
        F()
        result.append('F')
    elif color[5][0][0]==color[5][0][1]==color[5][0][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==color[2][1][2]==color[2][2][2] :
        if color[5][0][0]==color[5][0][1]==color[5][0][2]==color[3][1][1] :
            D()
            result.append('D')
            D()
```

```
result.append('D')
B()
result.append('B')
B()
result.append('B')
if color[5][1][1]==color[0][1][0] :
    d()
    result.append('d')
    temp = 'd'
else :
    D()
    result.append('D')
    temp = 'D'
L()
result.append('L')
r()
result.append('r')
B()
result.append('B')
B()
result.append('B')
l()
result.append('l')
R()
result.append('R')
if temp == 'd' :
    d()
    result.append('d')
else :
    D()
    result.append('D')
    B()
    result.append('B')
    B()
    result.append('B')
elif color[5][0][0]==color[5][0][1]==color[5][0][2]==color[0][1][1] :
    D()
    result.append('D')
    R()
    result.append('R')
    R()
    result.append('R')
    if color[2][1][1]==color[3][2][1] :
        D()
        result.append('D')
        temp = 'D'
    else :
        d()
        result.append('d')
        temp = 'd'
    B()
    result.append('B')
    f()
    result.append('f')
    R()
    result.append('R')
    R()
    result.append('R')
    b()
    result.append('b')
    F()
    result.append('F')
```

```
    if temp == 'D' :
        D()
        result.append('D')
    else :
        d()
        result.append('d')
    R()
    result.append('R')
    R()
    result.append('R')
elif color[5][0][0]==color[5][0][1]==color[5][0][2]==color[2][1][1] :
    d()
    result.append('d')
    L()
    result.append('L')
    L()
    result.append('L')
    if color[3][2][1]==color[0][1][1] :
        d()
        result.append('d')
        temp = 'd'
    else :
        D()
        result.append('D')
        temp = 'D'
    b()
    result.append('b')
    F()
    result.append('F')
    L()
    result.append('L')
    L()
    result.append('L')
    B()
    result.append('B')
    f()
    result.append('f')
    if temp == 'd' :
        d()
        result.append('d')
    else :
        D()
        result.append('D')
    L()
    result.append('L')
    L()
    result.append('L')
elif color[5][0][0]==color[5][0][1]==color[5][0][2]==color[5][1][1] :
    F()
    result.append('F')
    F()
    result.append('F')
    if color[3][1][1]==color[0][1][0] :
        D()
        result.append('D')
        temp = 'D'
    else :
        d()
        result.append('d')
        temp = 'd'
    l()
    result.append('l')
```

```
            R()
            result.append('R')
            F()
            result.append('F')
            F()
            result.append('F')
            L()
            result.append('L')
            r()
            result.append('r')
            if temp == 'D' :
                D()
                result.append('D')
            else :
                d()
                result.append('d')
            F()
            result.append('F')
            F()
            result.append('F')

elif not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[5][0][0]==color[5][0][1]=
=color[5][0][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[2][0][2]==
color[2][1][2]==color[2][2][2] and ((color[3][2][1]==color[5][1][1] and color[5][0][1]==color[3]
[1][1]) or (color[3][2][1]==color[2][1][1] and color[2][1][2]==color[3][1][1]) or (color[3][2][
1]==color[0][1][1] and color[0][1][0]==color[3][1][1])):
    if color[3][2][1]==color[5][1][1] and color[5][0][1]==color[3][1][1] :
        F()
        result.append('F')
        F()
        result.append('F')
    if color[3][1][1]==color[0][1][0] :
        D()
        result.append('D')
        temp = 'D'
    else :
        d()
        result.append('d')
        temp = 'd'
    l()
    result.append('l')
    R()
    result.append('R')
    F()
    result.append('F')
    F()
    result.append('F')
    L()
    result.append('L')
    r()
    result.append('r')
    if temp == 'D' :
        D()
        result.append('D')
    else :
        d()
        result.append('d')
    F()
    result.append('F')
    F()
    result.append('F')
elif color[3][2][1]==color[2][1][1] and color[2][1][2]==color[3][1][1] :
```

```
F()
result.append('F')
F()
result.append('F')
if color[3][1][1]==color[0][1][0] :
    D()
    result.append('D')
    temp = 'D'
else :
    d()
    result.append('d')
    temp = 'd'
l()
result.append('l')
R()
result.append('R')
F()
result.append('F')
F()
result.append('F')
L()
result.append('L')
r()
result.append('r')
if temp == 'D' :
    D()
    result.append('D')
else :
    d()
    result.append('d')
F()
    result.append('F')
    F()
    result.append('F')
elif color[3][2][1]==color[0][1][1] and color[0][1][0]==color[3][1][1] :
    R()
    result.append('R')
    R()
    result.append('R')
    if color[2][1][1]==color[3][2][1] :
        D()
        result.append('D')
        temp = 'D'
    else :
        d()
        result.append('d')
        temp = 'd'
    B()
    result.append('B')
    f()
    result.append('f')
    R()
    result.append('R')
    R()
    result.append('R')
    b()
    result.append('b')
    F()
    result.append('F')
    if temp == 'D' :
        D()
        result.append('D')
```

```
        else :
            d()
            result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')

if color[3][2][0]==color[3][2][2]==color[2][1][1]:
    D()
    result.append('D')
elif color[3][2][0]==color[3][2][2]==color[0][1][1]:
    d()
    result.append('d')
elif color[3][2][0]==color[3][2][2]==color[5][1][1]:
    D()
    result.append('D')
    D()
    result.append('D')

if (color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[0][0][0]==color[0][1][0]==colo
r[0][2][0] and not color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[5][0][0]==color
[5][0][1]==color[5][0][2]) or (color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][
2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==color[2][1][2]==color[2][2][2] an
d not color[5][0][0]==color[5][0][1]==color[5][0][2]) or (color[2][0][2]==color[2][1][2]==color
[2][2][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][2][0]==color[
3][2][1]==color[3][2][2] and not color[5][0][0]==color[5][0][1]==color[5][0][2]) or (color[5][0]
[0]==color[5][0][1]==color[5][0][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and
not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==color[2][1][2]==colo
r[2][2][2]) :
    if color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[0][0][0]==color[0][1][0]==c
olor[0][2][0] and not color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[5][0][0]==co
lor[5][0][1]==color[5][0][2] :
        if color[3][2][0]==color[3][2][1]==color[3][2][2]==color[3][1][1] :
            B()
            result.append('B')
            B()
            result.append('B')
            if color[5][1][1]==color[0][1][0] :
                d()
                result.append('d')
                temp = 'd'
            else :
                D()
                result.append('D')
                temp = 'D'
            L()
            result.append('L')
            r()
            result.append('r')
            B()
            result.append('B')
            B()
            result.append('B')
            l()
            result.append('l')
            R()
            result.append('R')
            if temp =='d' :
                d()
                result.append('d')
            else :
```

```
            D()
            result.append('D')
        B()
        result.append('B')
        B()
        result.append('B')
    elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[0][1][1] :
        d()
        result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')
        if color[2][1][1]==color[3][2][1] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        B()
        result.append('B')
        f()
        result.append('f')
        R()
        result.append('R')
        R()
        result.append('R')
        b()
        result.append('b')
        F()
        result.append('F')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[2][1][1] :
        D()
        result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
        if color[3][2][1]==color[0][1][1] :
            d()
            result.append('d')
            temp = 'd'
        else :
            D()
            result.append('D')
            temp = 'D'
        b()
        result.append('b')
        F()
        result.append('F')
```

```
        L()
        result.append('L')
        L()
        result.append('L')
        B()
        result.append('B')
        f()
        result.append('f')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[5][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        l()
        result.append('l')
        R()
        result.append('R')
        F()
        result.append('F')
        F()
        result.append('F')
        L()
        result.append('L')
        r()
        result.append('r')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        F()
        result.append('F')
        F()
        result.append('F')
    elif color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[5][0][0]==color[5][0][1]==color[5][0][2] :
        if color[0][0][0]==color[0][1][0]==color[0][2][0]==color[3][1][1] :
            D()
```

```
result.append('D')
B()
result.append('B')
B()
result.append('B')
if color[5][1][1]==color[0][1][0] :
    d()
    result.append('d')
    temp = 'd'
else :
    D()
    result.append('D')
    temp = 'D'
L()
result.append('L')
r()
result.append('r')
B()
result.append('B')
B()
result.append('B')
l()
result.append('l')
R()
result.append('R')
if temp == 'd' :
    d()
    result.append('d')
else :
    D()
    result.append('D')
    B()
    result.append('B')
    B()
    result.append('B')
elif color[0][0][0]==color[0][1][0]==color[0][2][0]==color[0][1][1] :
    R()
    result.append('R')
    R()
    result.append('R')
    if color[2][1][1]==color[3][2][1] :
        D()
        result.append('D')
        temp = 'D'
    else :
        d()
        result.append('d')
        temp = 'd'
    B()
    result.append('B')
    f()
    result.append('f')
    R()
    result.append('R')
    R()
    result.append('R')
    b()
    result.append('b')
    F()
    result.append('F')
    if temp == 'D' :
        D()
```

```
            result.append('D')
        else :
            d()
            result.append('d')
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[0][0][0]==color[0][1][0]==color[0][2][0]==color[2][1][1] :
        D()
        result.append('D')
        D()
        result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
        if color[3][2][1]==color[0][1][1] :
            d()
            result.append('d')
            temp = 'd'
        else :
            D()
            result.append('D')
            temp = 'D'
        b()
        result.append('b')
        F()
        result.append('F')
        L()
        result.append('L')
        L()
        result.append('L')
        B()
        result.append('B')
        f()
        result.append('f')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[0][0][0]==color[0][1][0]==color[0][2][0]==color[5][1][1] :
        d()
        result.append('d')
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
```

```
            l()
            result.append('l')
            R()
            result.append('R')
            F()
            result.append('F')
            F()
            result.append('F')
            L()
            result.append('L')
            r()
            result.append('r')
            if temp == 'D' :
                D()
                result.append('D')
            else :
                d()
                result.append('d')
            F()
            result.append('F')
            F()
            result.append('F')
    elif color[2][0][2]==color[2][1][2]==color[2][2][2] and not color[0][0][0]==color[0][1][0]=
=color[0][2][0] and not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[5][0][0]==
color[5][0][1]==color[5][0][2] :
        if color[2][0][2]==color[2][1][2]==color[2][2][2]==color[3][1][1] :
            d()
            result.append('d')
            B()
            result.append('B')
            B()
            result.append('B')
if color[5][1][1]==color[0][1][0] :
    d()
    result.append('d')
    temp = 'd'
else :
    D()
    result.append('D')
    temp = 'D'
L()
result.append('L')
r()
result.append('r')
B()
result.append('B')
B()
result.append('B')
l()
result.append('l')
R()
result.append('R')
if temp == 'd' :
    d()
    result.append('d')
else :
    D()
    result.append('D')
B()
result.append('B')
B()
result.append('B')
```

```
elif color[2][0][2]==color[2][1][2]==color[2][2][2]==color[0][1][1] :
    D()
    result.append('D')
    D()
    result.append('D')
    R()
    result.append('R')
    R()
    result.append('R')
    if color[2][1][1]==color[3][2][1] :
        D()
        result.append('D')
        temp = 'D'
    else :
        d()
        result.append('d')
        temp = 'd'
    B()
    result.append('B')
    f()
    result.append('f')
    R()
    result.append('R')
    R()
    result.append('R')
    b()
    result.append('b')
    F()
    result.append('F')
    if temp == 'D' :
        D()
        result.append('D')
    else :
        d()
        result.append('d')
    R()
    result.append('R')
    R()
    result.append('R')
elif color[2][0][2]==color[2][1][2]==color[2][2][2]==color[2][1][1] :
    L()
    result.append('L')
    L()
    result.append('L')
    if color[3][2][1]==color[0][1][1] :
        d()
        result.append('d')
        temp = 'd'
    else :
        D()
        result.append('D')
        temp = 'D'
    b()
    result.append('b')
    F()
    result.append('F')
    L()
    result.append('L')
    L()
    result.append('L')
    B()
    result.append('B')
```

```
        f()
        result.append('f')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[2][0][2]==color[2][1][2]==color[2][2][2]==color[5][1][1] :
        D()
        result.append('D')
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        l()
        result.append('l')
        R()
        result.append('R')
                F()
                result.append('F')
                F()
                result.append('F')
                L()
                result.append('L')
                r()
                result.append('r')
                if temp == 'D' :
                    D()
                    result.append('D')
                else :
                    d()
                    result.append('d')
                F()
                result.append('F')
                F()
                result.append('F')
    elif color[5][0][0]==color[5][0][1]==color[5][0][2] and not color[0][0][0]==color[0][1][0]=
=color[0][2][0] and not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[2][0][2]==
color[2][1][2]==color[2][2][2] :
        if color[5][0][0]==color[5][0][1]==color[5][0][2]==color[3][1][1] :
            D()
            result.append('D')
            D()
            result.append('D')
            B()
            result.append('B')
            B()
            result.append('B')
            if color[5][1][1]==color[0][1][0] :
```

```
            d()
            result.append('d')
            temp = 'd'
        else :
            D()
            result.append('D')
            temp = 'D'
        L()
        result.append('L')
        r()
        result.append('r')
        B()
        result.append('B')
        B()
        result.append('B')
        l()
        result.append('l')
        R()
        result.append('R')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        B()
        result.append('B')
        B()
        result.append('B')
    elif color[5][0][0]==color[5][0][1]==color[5][0][2]==color[0][1][1] :
        D()
        result.append('D')
        R()
        result.append('R')
        R()
        result.append('R')
        if color[2][1][1]==color[3][2][1] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        B()
        result.append('B')
        f()
        result.append('f')
        R()
        result.append('R')
        R()
        result.append('R')
        b()
        result.append('b')
        F()
        result.append('F')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
```

```
        R()
        result.append('R')
        R()
        result.append('R')
    elif color[5][0][0]==color[5][0][1]==color[5][0][2]==color[2][1][1] :
        d()
        result.append('d')
        L()
        result.append('L')
        L()
        result.append('L')
        if color[3][2][1]==color[0][1][1] :
            d()
            result.append('d')
            temp = 'd'
        else :
            D()
            result.append('D')
            temp = 'D'
        b()
        result.append('b')
        F()
        result.append('F')
        L()
        result.append('L')
        L()
        result.append('L')
        B()
        result.append('B')
        f()
        result.append('f')
        if temp == 'd' :
            d()
            result.append('d')
        else :
            D()
            result.append('D')
        L()
        result.append('L')
        L()
        result.append('L')
    elif color[5][0][0]==color[5][0][1]==color[5][0][2]==color[5][1][1] :
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        l()
        result.append('l')
        R()
        result.append('R')
        F()
        result.append('F')
        F()
        result.append('F')
```

```
            L()
            result.append('L')
            r()
            result.append('r')
            if temp == 'D' :
                D()
                result.append('D')
            else :
                d()
                result.append('d')
            F()
            result.append('F')
            F()
            result.append('F')

elif not color[3][2][0]==color[3][2][1]==color[3][2][2] and not color[5][0][0]==color[5][0][1]=
=color[5][0][2] and not color[0][0][0]==color[0][1][0]==color[0][2][0] and not color[2][0][2]==
color[2][1][2]==color[2][2][2] and ((color[3][2][1]==color[5][1][1] and color[5][0][1]==color[3]
[1][1]) or (color[3][2][1]==color[2][1][1] and color[2][1][2]==color[3][1][1]) or (color[3][2][
1]==color[0][1][1] and color[0][1][0]==color[3][1][1])):
    if color[3][2][1]==color[5][1][1] and color[5][0][1]==color[3][1][1] :
        F()
        result.append('F')
        F()
        result.append('F')
        if color[3][1][1]==color[0][1][0] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        l()
        result.append('l')
        R()
        result.append('R')
        F()
        result.append('F')
        F()
        result.append('F')
        L()
        result.append('L')
        r()
        result.append('r')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        F()
        result.append('F')
        F()
        result.append('F')
elif color[3][2][1]==color[2][1][1] and color[2][1][2]==color[3][1][1] :
    F()
    result.append('F')
    F()
    result.append('F')
    if color[3][1][1]==color[0][1][0] :
        D()
```

```
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        l()
        result.append('l')
        R()
        result.append('R')
        F()
        result.append('F')
        F()
        result.append('F')
        L()
        result.append('L')
        r()
        result.append('r')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        F()
        result.append('F')
        F()
        result.append('F')
    elif color[3][2][1]==color[0][1][1] and color[0][1][0]==color[3][1][1] :
        R()
        result.append('R')
        R()
        result.append('R')
        if color[2][1][1]==color[3][2][1] :
            D()
            result.append('D')
            temp = 'D'
        else :
            d()
            result.append('d')
            temp = 'd'
        B()
        result.append('B')
        f()
        result.append('f')
        R()
        result.append('R')
        R()
        result.append('R')
        b()
        result.append('b')
        F()
        result.append('F')
        if temp == 'D' :
            D()
            result.append('D')
        else :
            d()
            result.append('d')
        R()
        result.append('R')
        R()
```

```
        result.append('R')

if color[3][2][0]==color[3][2][1]==color[3][2][2]==color[2][1][1] :
    D()
    result.append('D')
elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[0][1][1] :
    d()
    result.append('d')
elif color[3][2][0]==color[3][2][1]==color[3][2][2]==color[5][1][1] :
    D()
    result.append('D')
    D()
    result.append('D')

print(result)



# for Raspberry pi

CW = 0     # Clockwise Rotation
CCW = 1    # Counterclockwise Rotation
SPR = 50   # Steps per Revolution (360 / 1.8)

# Defining pin numbers

STEP_L = 23
DIR_L = 24

STEP_R = 12
DIR_R = 16

STEP_B = 20
DIR_B = 21

STEP_F = 10
DIR_F = 9

STEP_D = 19
DIR_D = 26

# Defining MODE (Microstep Resolution GPIO pins)

MODE_L_MOTOR = (14, 15, 18)
MODE_R_MOTOR = (25, 8, 7)
MODE_B_MOTOR = (2, 3, 4)
MODE_F_MOTOR = (17, 27, 22)
MODE_D_MOTOR = (5, 6, 13)

sleep(5)

i=0
for i in range (len(result)) :
    if result[i]=='L' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_L, GPIO.OUT)
        GPIO.setup(STEP_L, GPIO.OUT)
        GPIO.output(DIR_L, CW)
        GPIO.setup(MODE_L_MOTOR, GPIO.OUT)
        GPIO.output(MODE_L_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001
```

```
        x=0
        for x in range (step_count) :
            GPIO.output(STEP_L, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_L, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='l' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_L, GPIO.OUT)
        GPIO.setup(STEP_L, GPIO.OUT)
        GPIO.output(DIR_L, CCW)
        GPIO.setup(MODE_L_MOTOR, GPIO.OUT)
        GPIO.output(MODE_L_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_L, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_L, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='R' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_R, GPIO.OUT)
        GPIO.setup(STEP_R, GPIO.OUT)
        GPIO.output(DIR_R, CW)
        GPIO.setup(MODE_R_MOTOR, GPIO.OUT)
        GPIO.output(MODE_R_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_R, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_R, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='r' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_R, GPIO.OUT)
        GPIO.setup(STEP_R, GPIO.OUT)
        GPIO.output(DIR_R, CCW)
        GPIO.setup(MODE_R_MOTOR, GPIO.OUT)
        GPIO.output(MODE_R_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001
```

```
        x=0
        for x in range (step_count) :
            GPIO.output(STEP_R, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_R, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='B' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_B, GPIO.OUT)
        GPIO.setup(STEP_B, GPIO.OUT)
        GPIO.output(DIR_B, CW)
        GPIO.setup(MODE_B_MOTOR, GPIO.OUT)
        GPIO.output(MODE_B_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_B, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_B, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='b' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_B, GPIO.OUT)
        GPIO.setup(STEP_B, GPIO.OUT)
        GPIO.output(DIR_B, CCW)
        GPIO.setup(MODE_B_MOTOR, GPIO.OUT)
        GPIO.output(MODE_B_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_B, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_B, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='F' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_F, GPIO.OUT)
        GPIO.setup(STEP_F, GPIO.OUT)
        GPIO.output(DIR_F, CW)
        GPIO.setup(MODE_F_MOTOR, GPIO.OUT)
        GPIO.output(MODE_F_MOTOR, (0, 0, 0))

        step_count = SPR
```

```
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_F, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_F, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='f' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_F, GPIO.OUT)
        GPIO.setup(STEP_F, GPIO.OUT)
        GPIO.output(DIR_F, CCW)
        GPIO.setup(MODE_F_MOTOR, GPIO.OUT)
        GPIO.output(MODE_F_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_F, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_F, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='D' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_D, GPIO.OUT)
        GPIO.setup(STEP_D, GPIO.OUT)
        GPIO.output(DIR_D, CW)
        GPIO.setup(MODE_D_MOTOR, GPIO.OUT)
        GPIO.output(MODE_D_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_D, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_D, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='d' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_D, GPIO.OUT)
        GPIO.setup(STEP_D, GPIO.OUT)
        GPIO.output(DIR_D, CCW)
        GPIO.setup(MODE_D_MOTOR, GPIO.OUT)
        GPIO.output(MODE_D_MOTOR, (0, 0, 0))
```

```
        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_D, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_D, GPIO.LOW)
            sleep(delay)

        sleep(5)

GPIO.cleanup()
```