FINAL REPORT

Of

RUBIK'S CUBE SOLVER

School of Computer Science and Engineering

Lovely Professional University.

Phagwara, Punjab (India).

INT 404 – ARTIFICIAL INTELLIGENCE

Submitted to:

Mrs. Shabnam

GitHub link: -

https://github.com/SANIDHYADASH/INT404-AI-Rubik-s-Cube-Solver

Submitted By:

Roll no	Registration No	Name of the Student	Marks	Signature
35	11802909	Utkarsh Patel		
36	11802843	Md. Saqulain		
39	11802151	Sanidhya Dash		
38	11802150	Debiprasad Sahoo		

ABSTRACT

➔ Rubik’s Cube is a widely popular mechanical puzzle that has attracted attention around the world because of its unique characteristics. As a classic brain-training toy well known to the public, Rubik’s Cube was used for scientific research and technology development by many scholars. This paper provides a basic understanding of the Rubik’s Cube and shows its mechanical art from the aspects of origin and development, characteristics, research status and especially its mechanical engineering design, as well as making a vision for the application in mechanism. First, the invention and origin of Rubik’s Cube are presented, and then the special characteristics of the cube itself are analysed. After that, the present researches of Rubik’s Cube are reviewed in various disciplines at home and abroad, including the researches of Rubik’s Cube scientific metaphors, reduction algorithms, characteristic applications, and mechanism issues. Finally, the applications and prospects of Rubik’s Cube in the field of mechanism are discussed.

RELATED WORK

- Morwen B. Thistlethwaite is a mathematician who devised a clever algorithm for solving the Rubik's Cube in remarkably few moves. It is a rather complicated method, and therefore cannot be memorised. It is only practical for computers and not for humans. This algorithm is rather important from a theoretical standpoint however, as it has long been the method with the fewest number of moves.

  Thistlethwaite's method differs from layer algorithms and corners first algorithms in that it does not place pieces in their correct positions one by one. Instead it works on all the pieces at the same time, restricting them to fewer and fewer possibilities until there is only one possible position left for each piece and the cube is solved.

The way it does this is by first doing a few moves until a position arises that can be solved without using quarter turns of the U and D faces (though half turns of U and D are still needed). It then proceeds to solve the cube without using U or D quarter turns by first moving to a position that does not need quarter turns of the F, B faces either. With these further restrictions a position is arrived at that does not need any quarter turns at all, and can hence be solved by half turns only. The cube then indeed gets solved using half turns only.

- Ryan Heise has developed a way of solving the cube based on Thistlethwaite's algorithm. He splits stages 2, 3 and 4 into two steps each (corners and edges separately) in order to make it possible for a human being to memorise.
- An algorithm for finding optimal solutions for Rubik's Cube was published in 1997 by Richard Korf. While, it had been known since 1995 that 20 was a lower bound on the number of moves for the solution in the worst case, it was proved in 2010 through extensive computer calculations that no configuration requires more than 20 moves. Thus 20 is a sharp upper bound on the length of optimal solutions. This number is known as **God's number**.

IMPLEMENTATION

➔ A Rubik's Cube algorithm is an operation on the puzzle which reorients its pieces in a certain way. Mathematically the Rubik's Cube is a permutation group: an ordered list, with 54 fields with 6*9 values (colours) on which we can apply operations (basic face rotations, cube turns and the combinations of these) which reorient the permutation group according to a pattern.

➔ The names of the facelet positions of the cube (letters stand for Up, Left, Front, Right, Back, and Down):

```
|***********|
|*U1**U2**U3*|
|***********|
```

```
            |*U4**U5**U6*|
            |************|
            |*U7**U8**U9*|
            |************|
************|************|************|************
*L1**L2**L3*|*F1**F2**F3*|*R1**R2**R3*|*B1**B2**B3*
************|************|************|************
*L4**L5**L6*|*F4**F5**F6*|*R4**R5**R6*|*B4**B5**B6*
************|************|************|************
*L7**L8**L9*|*F7**F8**F9*|*R7**R8**R9*|*B7**B8**B9*
************|************|************|************
            |************|
            |*D1**D2**D3*|
            |************|
            |*D4**D5**D6*|
            |************|
            |*D7**D8**D9*|
            |************|
```

A cube definition string "UBL..." means that in position U1 we have the U-colour, in position U2 we have the B-colour, in position U3 we have the L colour etc. according to the order U1, U2, U3, U4, U5, U6, U7, U8, U9, R1, R2, R3, R4, R5, R6, R7, R8, R9, F1, F2, F3, F4, F5, F6, F7, F8, F9, D1, D2, D3, D4, D5, D6, D7, D8, D9, L1, L2, L3, L4, L5, L6, L7, L8, L9, B1, B2, B3, B4, B5, B6, B7, B8, B9.

So, for example, a definition of a solved cube would be UUUUUUUUURRRRRRRRRFFFFFFFFFDDDDDDDDDLLLLLLLLLBBBBBBBBB

Solution string consists of space-separated parts, each of them represents a single move:

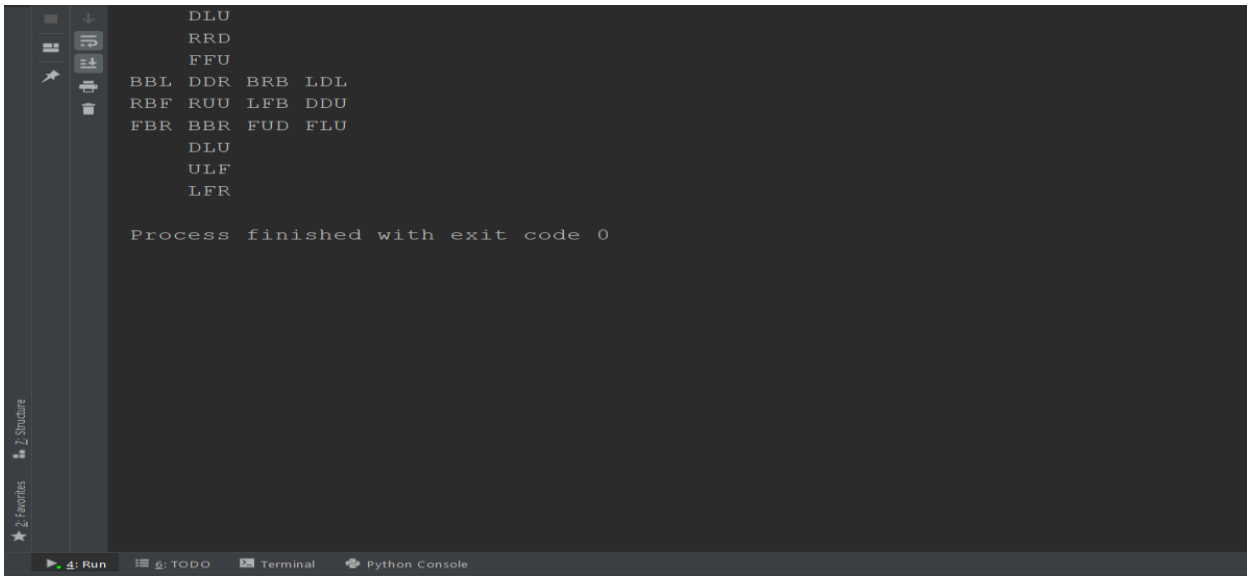
- A single letter by itself means to turn that face clockwise 90 degrees.
- A letter followed by an apostrophe means to turn that face counter clockwise 90 degrees.
- A letter with the number 2 after it means to turn that face 180 degrees.

e.g. **R U R' U R U2 R' U**

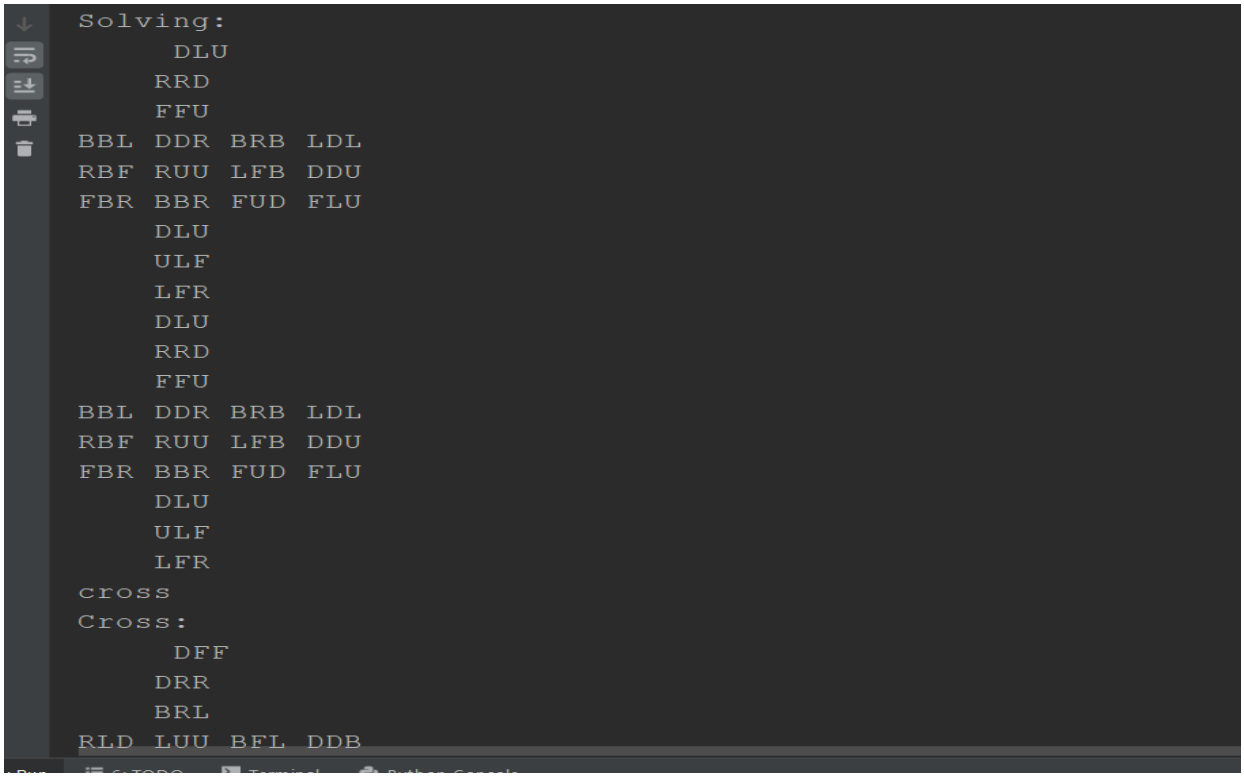
RESULT / OUTPUT

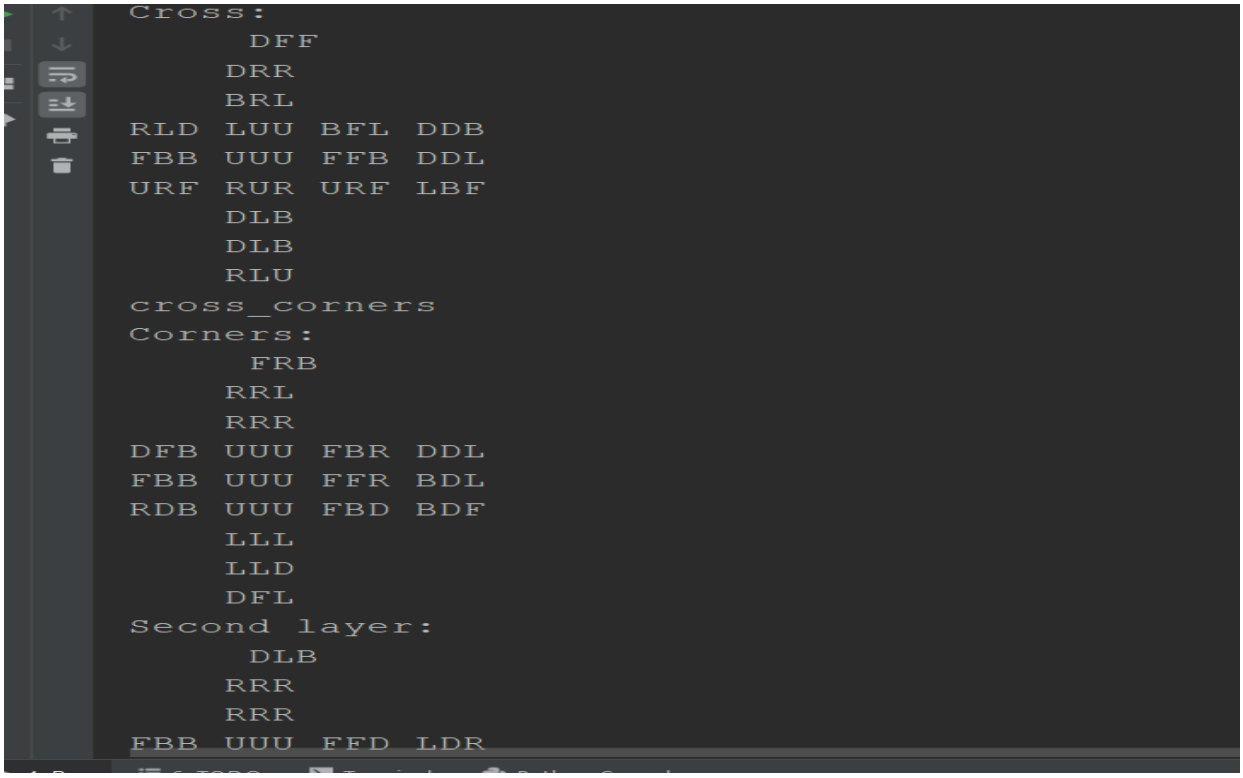
→On executing the different files of the complete program for Rubik's cube solving, the result that will be obtained are:

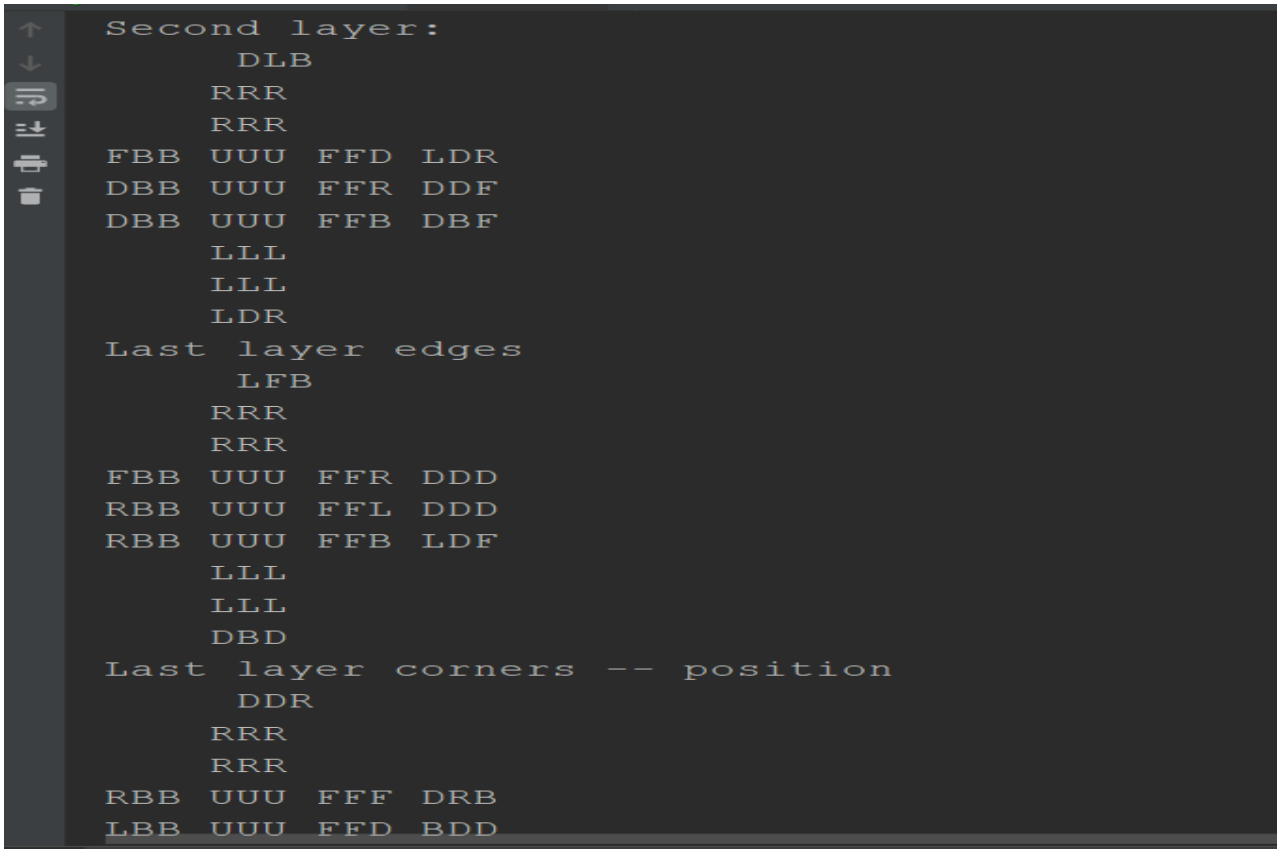
**main.py:



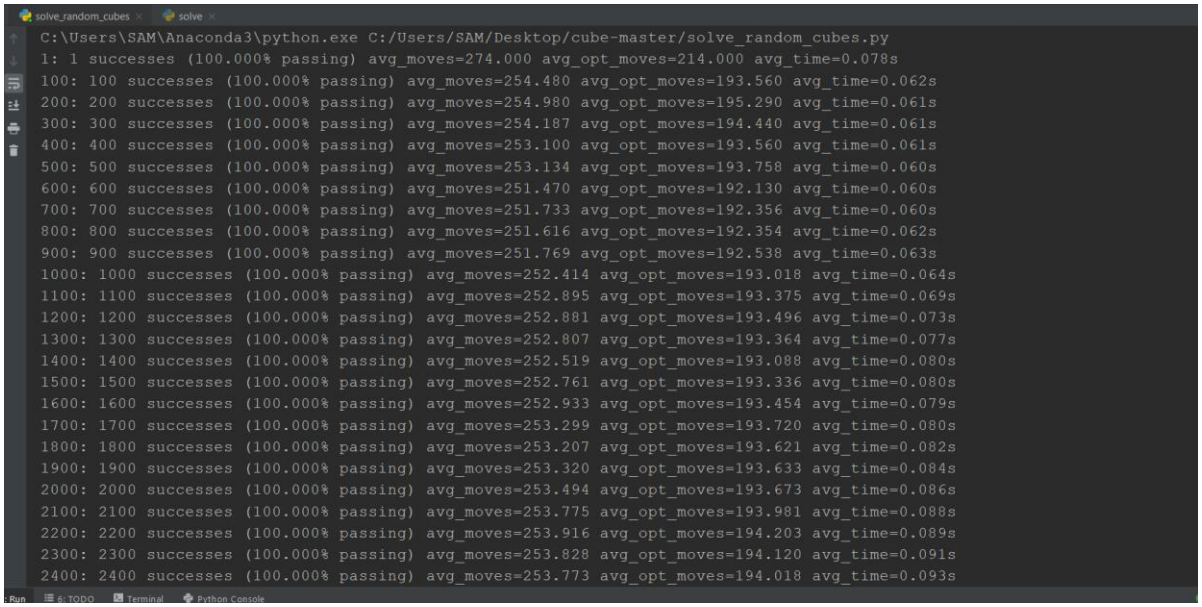
**CubeSolve.py:

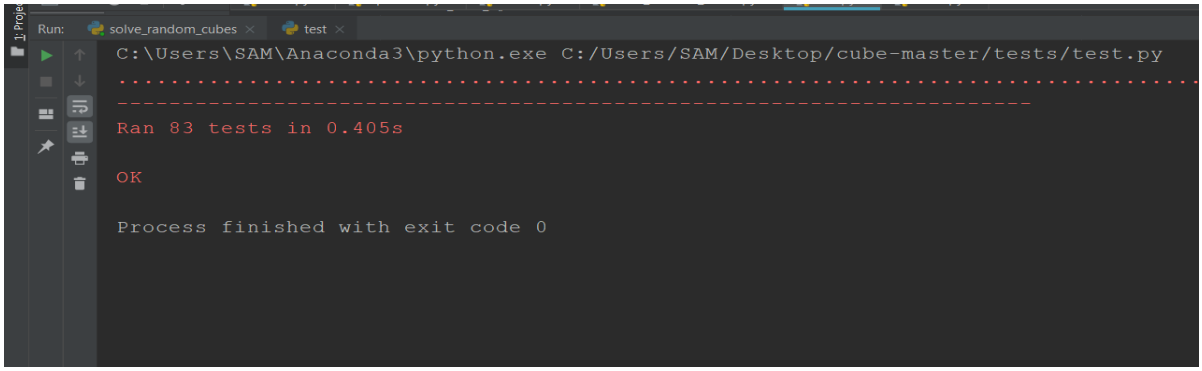

Solving:
DLU
RRD
FFU
BBL DDR BRB LDL
RBF RUU LFB DDU
FBR BBR FUD FLU
DLU
ULF
LFR
DLU
RRD
FFU
BBL DDR BRB LDL
RBF RUU LFB DDU
FBR BBR FUD FLU
DLU
ULF
LFR
cross
Cross:
DFF
DRR
BRL
RLD LUU BFL DDB


Cross:
DFF
DRR
BRL
RLD LUU BFL DDB
FBB UUU FFB DDL
URF RUR URF LBF
DLB
DLB
RLU
cross_corners
Corners:
FRB
RRL
RRR
DFB UUU FBR DDL
FBB UUU FFR BDL
RDB UUU FBD BDF
LLL
LLD
DFL
Second layer:
DLB
RRR
RRR
FBB UUU FFD LDR



**random_cubes.py:





IMPORTANT LIBRARIES USED

NumPy → Used for using rotation functions

Random → Used for scrambling the cube for random colour positions

Sys → System-specific parameters and functions. This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter.

Time → It provides many ways of representing time in code, such as objects, numbers, and strings. It also provides functionality other than representing time, like waiting during code execution and measuring the efficiency of your code.

Textwrap → It can be used for wrapping and formatting of plain text and it provides formatting of text by adjusting the line breaks in the input paragraph.

TEAM RESPONSIBILITIES

Utkarsh -> Make the representation of the cube for solving

Md. Saqulain -> Use the algorithm for solving the Rubik's cube

Sanidhya Dash -> Use the algorithm for solving the Rubik's cube

Debiprasad Sahoo -> Use the algorithm for solving the Rubik's cube

REFERENCES

https://ruwix.com/the-rubiks-cube/algorithm/

https://rubiks-cube-solver.com/

https://fulmicoton.com/posts/rubix/

https://towardsdatascience.com/learning-to-solve-a-rubiks-cube-from-scratch-using-reinforcement-learning-381c3bac5476

https://en.wikipedia.org/wiki/Rubik%27s_Cube#Conception_and_development

https://www.jaapsch.net/puzzles/thistle.htm