Table of Contents

Introduction to the project

What we want to do.

Why did we choose this idea?

Researched information

Main source (solvethecube.com)

Notation

Inspired from program

The proposed solution

3 steps to run the program

User input

Solve

Return solution to user

Will use the wiki to complete this section

The actual solution

Classes and how they interact

Graphical interface explained

Screen shots of the final form

Dry-Run

Methodological run of the program with screenshots

Introduction to the project

What do we want to create?

We were asked to come up with a project idea that would have us use the Qt framework and its graphical user interface mechanisms as thoroughly as possible. The first few ideas that came to mind were game ideas because they are very demanding in GUI but we were not very enthused by the fact that we would have to be working with C++. So we thought about it a little further and the idea which came to us as the most interesting, that uses both GUI and C++ effectively, and that would produce a *useful* product is solving a Rubik’s Cube using a computer. The potential in that project lies in the fact that it would have to interact with the user in such a way that the latter can give the current status of their unsolved Cube and the program will solve the cube and output the series of moves to the user in a comprehensive way. The difficulty of this project lies in the implementation of the algorithmic hidden behind the resolution of the cube and using C++ lightens the amount of resources required to solve the cube, for this language handles RAM much better than Java, for instance, would.

Why did we choose this idea?

We