

Description:

This game is a **Quantum gates** themed variation of Candy Crush called QG Crusher. It is a minimum match-three puzzle game where the player can swap adjacent elements on the board to form a line (horizontal) of three (or more) gates to make an identity gate. Identity gate in the sense of getting a unitary action while aligning the different combination of the gates. When such a line is formed, the elements are eliminated (crushed) from the game board, and pieces above them fall to occupy the now empty spaces. Each completed line earns points, with more points awarded for constructing longer lines or multiple lines in the same turn. The goal of the game is to earn the required number of points before running out of turns. All of the user interface / animation code and graphics are provided to the players.

Metadata

Summary	The QG Crusher game has a logic that implements the Quantum gate decomposition themed with a minimum match of three gates which results in Identity gate and disappear from screen similar to Candy Crush. The starting materials provide a polished game interface that is platform independent.
Audience	Anyone can Play this game who has a little bit introduction on Basics of Quantum information and Quantum gates.
Difficulty	Players found this game to be moderately difficult because it was the first game where they were required to calculate the matrix multiplications which results in identity gate.
Topics	The game focuses on Quantum Gates and Decomposition of the Quantum Gates.
Strengths	The provided code includes a polished user interface, including animation, Players also learn the importance of Quantum Gates and their properties like reversibility and unitary operation. They can also learn about construction of one gate using a different gates, which leads to the optimisation for hardware implementation in Quantum gates.
Weaknesses	In this game we only using 5 Gates, but actually there are nearly 16 gates are there, due to complexity of finding out the number different combinations where we can get the Identity gate. We neglected some basic gates like Rx, Ry, Rz gates. The game only check for the horizontal combinations Some bugs are there in the game, like getting the same combination of the gates because of random function limitation.
Dependencies	The user interface is built using our Simple Graphics module, which is a wrapper for Python's standard TKInter graphics module (and is included in the materials we have provided). The provided code executes successfully with Python 3 on Windows, MacOS and Linux.
Variants	We chose to provide players with the code needed to remove the game pieces when a match is made because there are actually many different cases that need to be considered for lines of 4 pieces, and lines, L-shaped groups and T-

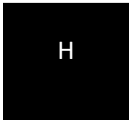




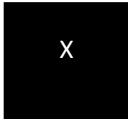




shaped groups of 5 pieces. The provided code could be reworked so that players are required to write this code to make the game more difficult.

Candy Crush includes many elements that are not included in our game that make the gameplay more interesting (and the game more complex to implement). One could introduce any number of these elements to increase the difficulty of the game, or to vary exactly what players need to do to complete it from term to term. Examples of variations that could be implemented include:

- A different special piece could be when a line of 4 identical pieces is formed. In Candy Crush this piece clears one row or one column in the board. Adding such a piece would provide additional opportunities for players to write functions that manipulate a 2D list.
- Block spaces could be added to the board. This would make the gameplay more interesting, and would also increase the complexity of the functions which cause the pieces to fall down into cleared space.
- Pieces that are "locked" and must be "released" before they can be cleared by subsequent moves.
- Pieces that alternate between two different symbols on even and odd turns.
- Conveyor belts that move the pieces sitting on them each turn.

Some Examples how game works

5-elimination

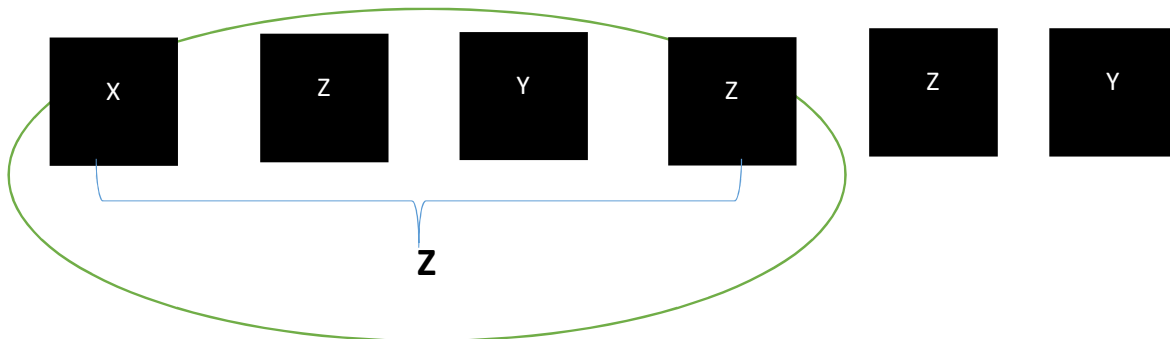
A					
B					

In above case, In row **A** we see there **X** gates are there but their result is not an identity gate. but we can see that in row **B** we have an **X**, so here we swap the both **Y,X** so that **A** row becomes **HXXXX**. Then action of 4 **XXXX** will lead to an Identity thus, the blocks will be blasted.

This is not the main theme of our game, our main goal is, build the Identity gate from using different gates, we have tested nearly 3125 test cases where we are getting the Identity gate using different gates.

Here we can see some Examples of how we built **X**, **Y**, **Z** gates from using different combinations of **X**, **Y**, **Z**.

Here I am taking an example where I am going to get Z gate with a different combination of XYZ and S gate.



In the above case we can see no possible for getting an Identity gate, but we go deep down in gate decomposition we know we can built each gate with different combination of other gates

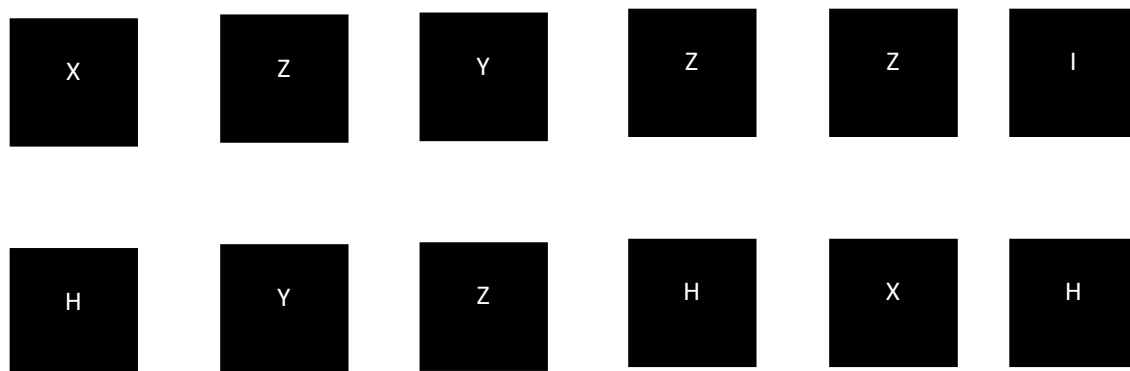
Suppose above I am looking Z gate, because its occurrence is high, although X is also same, but I am choosing Z.

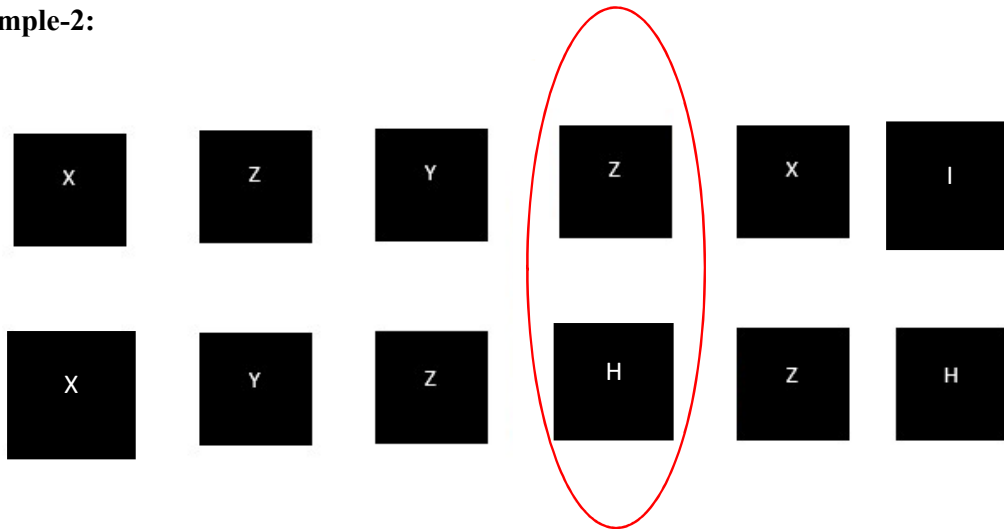
Now Z can be represented as $Z = XZYX$, so the first 4 block will give me a Z, so I need one more to make it Identity gate.

Sometimes a phase also will plays a major role, because some decompositions will add some phase to the system, then we need to consider the phase gate also. But we are considering that Phase gate for our future work.

Now we look at another scenario, where we have the option of eliminating by 3 but, it doesn't support or satisfies the unitary requirement.

Example-1:



Example-2:

But you guys will have doubt , why I has to do all these calculations and , do consider the phase and all other things ,her I can simply swap the **Z** and **H** to eliminate the those 3 Z boxes. But this won't work here because it is not ordinary Candy Crush game. Even though it is a minimum 3 match elimination game, but you have keep the unitary action/ getting Identity gate, in mind.

4-elimination

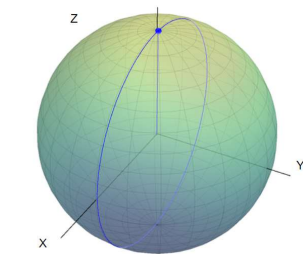
The above combination will results in the Identity gate.

Lets

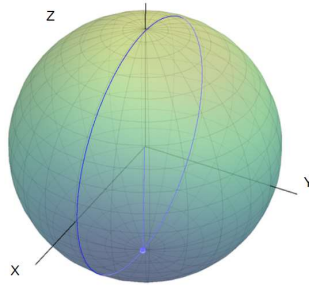
Likewise computed the all possible combinations for the **Five gates** that which we are going to use

Quantum Scenario:

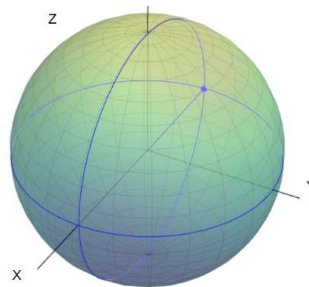
Now let's see how these Quantum gates are actually work in Quantum world. Suppose you have a Qubit_0 which is in state of $|0\rangle$, now let's see how these gates operate.



Above is at $|0\rangle$, now we are going to apply **X** gate, then let's see how its changes.

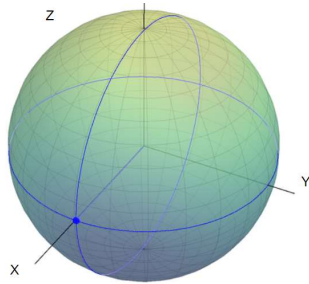


Now the Qubit is at position $|1\rangle$, now **H** gate acts on the current state.

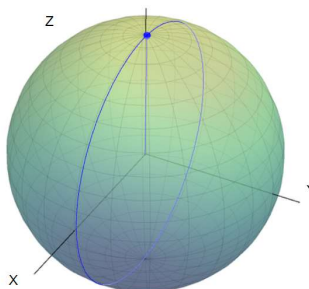


Qubit is in $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

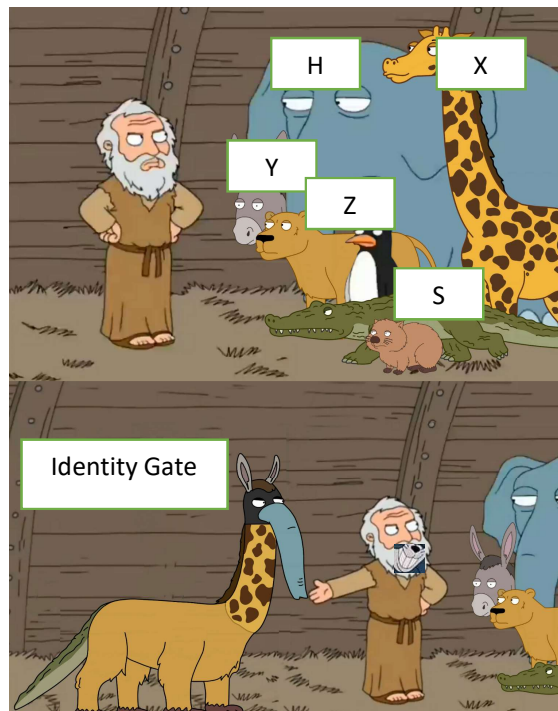
Now **Z** acts on the Present state and it transfer the present them to $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$



And finally **H** gate, which results or sets back to the initial and original state $|0\rangle$.



So finally after applying those sequence of gate we are again came back to the original position. So the action of the 4 gates resulted in the Unitary / Identity action. This is what we are implementing in the game.



Game Rules:

- Click on Play! to start playing the game
- **Objective:** Usually, goal is to clear as many rows as possible in as few moves as possible.
- You can clear quantum gates from the board by lining up three or more (combination of 2 gates are restricted because of the high probability of occurrence) in a row, which is accomplished by swapping quantum gates in order to create Identity gate.
- Whenever the gates will create Identity gate in the horizontal combination, the score is given according to the number of gates making the combination as follows:
 1. For 3,4,5,6,7,8 gate: score is 10,20,30,40,50,60 respectively
 2. You need to achieve the given target in the given number of moves.

Future Work:

As earlier in one scenario, I had mentioned about the phase gate. In initial levels we had kept only the combinations which doesn't require any phase gates, we are thinking to escalate our game by adding some gates, then our game will contain a total of 16 gates. As making a 16x16 game board is much difficult, so we keep board size only up to 8 or 10, because if the size of the board is more than the combinations we are going to calculate will become more exhaustive, So in future we will add more features, if combination of 2 gates giving another gate we will replace that 2 gates by a single gate. And we also add some boost up features.