**Plan**

Rules:

Tic Tac Toe has the following rules:

* Two players exist
* Players must alternate turns
* Each player has an assigned symbol – either a circle or cross
* The starting player is always assigned the cross symbol
* A grid exists
* At each turn, the respective player must place their assigned symbol in an empty cell (of their choice) on the grid
* A player wins if three of his symbols align in a straight line. This could be either horizontally, vertically, or diagonally.
* A game draws if all cells are filled, and neither player has won

Planning Decisions:

1. *Variables and constants*

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Usage |
| grid | GameGrid (custom type) | Game grid and values of cells will be stored in this variable. |
| grid\_dimension | int | Static constant, part of GameGrid class, which will define the dimensions of the grid. Initially, this will be set to , in order to generate a grid. Grids of variable sizes can be generated by changing the value of this constant |
|  |  |  |

1. *Use of data structures and files*

I will be using a personal data structure, GameGrid, which will be a class that deals with game mechanics. Internally, it will use a list data structure which will store the values for each cell of the grid. The list will only be one-dimensional, as it will use column-based ordering in order to map the two-dimensional game grid into a one-dimensional list.

A basic list of reasons why one-dimensional lists will be used instead:

* + This approach is more performant, as less objects have to be instantiated.
  + The game grid is very small, so removing the need for extra lists reduces the complexity of the design.
  + In C++ the use of two-dimensional arrays is discouraged. As a consequence, I prefer using one-dimensional array mappings.

1. *User interface*

Initially, I considered a command-line interface design. After a brief moment of thought, I realised that this would not be very user-friendly, as the user would have to enter coordinates where they would like to place their symbol, in one form or another (they could also enter a cell index, however this would still involve a coordinate system).

Subsequently, I realised that a GUI would be a much better design choice. I briefly researched which would be the most suitable GUI framework to use.

I considered the following frameworks:

* PyQt
* PySide
* PyGTK

PyQt was my initial choice. I quickly realised, however, that my choice was quite expensive. PySide is quite similar to PyQt (but free), meaning that it was my next choice. Unfortunately, PySide only works up to Python 3.4, which did not satisfy my requirements.

I ended up settling for PyGTK, as this is quite a powerful framework, yet free, and works on my platform (Python 3.6.3).

My GUI will be quite a simple interface, similar to the Google Tic Tac Toe game, although

