Monopoly Group 3 Design Decisions Made Milestone2:

Authors:

- Sarah Chow 101143033
- Kyra Lothrop 101145872
- Bardia Parmoun 101143006
- Owen VanDusen 101152022

MVC Pattern:

- MVC Advantages: flexibility, re-use, multiple presentations of the same data and allowing the user to select what data to present during runtime.
- For all these reasons for user interaction with a monopoly game, the MVC pattern is used
    - **The Model:** BoardModel.java
        - This class handles the logic for the monopoly game
        - BoardView/Listener are BoardModel's only point of contact. BoardModel sends updates that are printed in BoardController if user input is required or printed in the view if no user input is required
    - **The View:** BoardView.java
        - Interface class which contains the method signatures that are implemented in the BoardListener class
    - **The Controller:** BoardController.java
        - This class prompts users for information and passes the information to the Board Model class

Abstract class - BoardCell:

- Abstract BoardCell.java contains the name, owner and cell type of the cell with getters and setters
- This contains the information relating to individual cells on a monopoly board
- Tax.java, Go.java and Property.java inherits from board cell to have tax, go or property classes with specific Monopoly attributes for example, to store a reward in the GO cell

BoardEvent and Board Frame:

- **Board Event**: BoardEvent encapsulates the information that the model passes to its views. The board event contains many constructors since events can come with different forms.
- **BoardFrame**: BoardFrame extends JFrame and contains the swing elements that the user can interact with to create a GUI-based board

Array List Usage:

- The views, cells and players in BoardModel are stored in arraylists to be able to remove and add to the arraylists
- The addition capabilities of an arraylist are needed to add many properties/go/tax locations to the board for the cells arraylist

Enum Usage:

- Enums are used for cell types and possible board statuses
- A main reason enums are used is to reduce the chance of a typo in the code and in turn the need to update the typo in multiple locations in the classes
- In using the enums for the statuses they can have a stringCommand to keep track of the player string commands. Using this, there is only one location in the code that contains the string of the possible player commands