//Name(s):
Maura Winstanley
Matthew Donovan
Gameplay requirements:

1. Our game is based on Candyland. The goal is to move from the start square to the end square faster than the other players.
2. The board is a path of squares with colors. The player advances by drawing a card when it is their turn and moving to the square indicated by the card. Cards will have colors and the player will generally move to the next given square with that color, but squares can also randomly have power-ups, and if a player lands on a square with a power-up they will get to use that power up on their next turn. Power-ups will be things like: advance 2 squares of that color or go back to the previous square of that color.
3. Our theme is candy, this will be reinforced by pastel colors and candy across the decades representing the lifelong treasure that candyland holds to americans.
4. The player will be able to acquire power-ups (or downs) that will change how they move. The power ups will be:
    - Move forward two squares of a given color
    - Move back to the previous square of a given color
    - Move forward to a given square color + 1 square
5. A power-up is acquired by landing on a PowerSquare and released when a player chooses to utilize it. A player can have at most one power-up at any given time. If they already have a power-up and land on another PowerSquare, the old power-up will remain in their possession.
6. There will also be different types of squares represented by a class. A derived Square type will contain power ups/power downs that the user can acquire and use at their will. The cards will be represented by classes. The base class will be the standard card containing only one square of a varying color.
7. The special event that occurs is a universal event that can randomly occur that will move every player back/forward one square.
8. We will allow up to 4 players. Any number of players between 0 and 4 will be permitted.
9. The basic computer strategy will be to draw the card and complete the move - given that the game is mostly luck-based instead of strategy-based. The only difference between a CPU and human player is that when a CPU player lands on a PowerSquare, it will use it on the next turn every time, while a player gets to choose when/if they want to use their power-up.
10. The beginning and end states will be the clearly defined start square and stop square.

Design patterns:

We will use a factory design pattern to create Players. Our design will be

GameWindow	**calls**

GameWindow()	Board::NewGame()
	Graph()::Display()

NewGame()	Square::Square()
	Player::Player()
	Graph()::Display()

TakeTurn()	Player::drawCard()
	Player::get_power_up()
	Player::move()

checkForWinner()	Player::get_location()
	Player::increment_wins()
	Graph()::Update()

get_players()	

Player- base player (CPU)	**calls**
Player()	-
get_location()	-
set_location()	

move()	Player::get_location()
get_wins()	-
increment_wins()	-
get_powerup()	Square::get_powerup()
set_powerup()	-
usePower()	Player::get_powerup()
drawCard()	Board::get_random_card()

Human- derived from Player	**calls**
Human()	-

get_icon_color()	-
set_icon_color()	

Square - basic square	calls
Square()	-
get_location()	
get_value()	

PowerSquare - advanced square	calls
Square()	-
get_powerup()	

Graph	**-**
Update()	CellWindow:: get_players() Player::get_wins()
Display()	CellWindow:: get_players() Player::get_wins()

By checkpoint 1, we plan to have:

- the ui complete excluding any minor tweaks we will make as we work
- the skeleton for all the classes we need (.h and .cpp files) and their functions
- the factory to create players
- the board displayed

A user will be able to click the new game button and be shown a randomized board, as well as choose a number of players and enter their information.