

The goal of this game was to allow players to choose their location, according to a 12-hour clock, which would ultimately determine the number of customers they attract leading to the amount of money they earn. The game ran for the whole 100 days of summer before coming to an end.

In the beginning, each of the players randomly selected their location and then started to play their strategies after the first five rounds. This was done to create a history of the players to allow them to identify the best path for their strategies. Each player was assigned a different strategy to improve their personal utility. The location and the utilities of each of the players are represented as a global variable that is constantly populated during every round played by each of the players.

The first strategy employed a **mixed strategy** for Candy (Player One), which is a probability distribution over the pure strategies. This tactic incorporates a certain amount of randomization amongst the several potential courses of action depending on a specific probability. Using a **mixed strategy** approach, player one (Candy) calculated the probability that the other two players, i.e Bob and Alice, would select a particular location based on their previous locations. This was accomplished with the use of imported libraries, such as statistics and random modules.

Using “statistics.mode”, the frequent location based on Bob and Alice's previous locations is picked as the next destination. The most frequent location is identified using the first 5 random values and then continues to change as new locations generated by Bob and Alice's strategy are taken into consideration when finding the most frequent locations. If there is any collocation between the two players, the choice() function was employed, which randomly selects an element from the supplied sequence (a range from 1 to 12) excluding the colocated value. Following the determination of Bob and Alice's positions, an equidistant location was selected by calculating the clockwise and anticlockwise distances. The larger distance was then considered for determining the equidistant location of Candy.

With this knowledge, Candy could choose a location that is equidistant from each of the other players. For example, based on Bob's list of prior locations, his next location could be 11 o'clock and similarly, Alice's location could be 8 o'clock. Once their placements are taken into account, Candy can put up her lemonade shop at 3 o'clock receiving the highest utility of 9.

The second strategy was designed for Bob (Player two) which used both a **cooperative** and **non-cooperative technique**. Initially, using a **cooperative strategy**, player two decided to collaborate with player one (Candy) to create a joint benefit by ensuring they were not co-located and thus maximising their profits. Based on Candy's current location, Bob chooses a location using the choice function(), which adds one of these values, i.e., 2, 3, or 4, at random to Candy's current location to minimise the distance between them. This is done to ensure that the revenues are shared between the two players. If they are at the same position, the greatest distance value, i.e, 6, is added to the current value of Candy to guarantee no collaboration occurs.

However, after running the game 100 times and comparing the utility values of each player, It became evident that transitioning Bob to a **non - cooperative approach** would provide him with a greater benefit. Therefore, player two decided to update his strategy and adopt a similar tactic to that of player one. Player two decided to use a **mixed strategy** to keep track of the position and utility of each player, including himself. Taking into consideration of the previous values of the location and utility, Bob choose the location that gave the highest payoff. This strategy compares the utilities of each of the players and the position of the maximum utility is retrieved. This position is further considered as the new location for Bob.

An experiment was carried out, as shown in the appendix, to confirm that updating the strategies was the rationale decision. Figure 1 depicts the results obtained by employing only a **cooperative technique**, whereas Figure 2 depicts the results obtained by utilising a **cooperative technique** followed by a **non-cooperative technique**. Bob had a persistent losing streak, shown by his previous utility values, and therefore Bob decided to deviate from the **cooperative strategy** and upgrade to a **non-cooperative approach** (after 20 iterations), which delivered a better benefit and improved his utility value.

The third strategy was implemented for Alice (Player three), who choose a location that would maximise her utility based on the most frequent locations of Bob and Candy. Using “statistics.mode”, the frequent location based on Bob and Candy's previous locations is picked as the next destination.

The most frequent location continues to change as new locations generated by Bob and Candy's strategy are taken into consideration when finding the most frequent locations. If there is any collocation between the two players, the choice() function was employed, which randomly selects an element from the supplied sequence (a range from 1 to 12) excluding the colocated value.

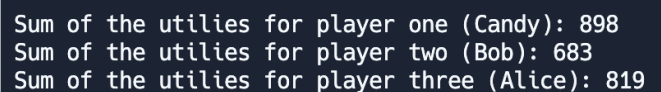
This strategy then checks if player two (Bob) and player one (Candy) have a difference of 4, if so, then Alice chooses a location that is equidistant from the players by adding 4 to her current value. This would lead to all the players having a utility value of 8. In this case, the players have no incentive to deviate and thus this can be considered a **Nash Equilibrium**. This would only be possible if players had a shared understanding of how the game is played and are rationale game players.

Nevertheless, if player two (Bob) and player one (Candy) are not 4 spots away, the strategy then checks the distance between the two players. If they are currently far apart, Alice would pick a location that is closer to either Bob or Candy by adding a 2 to the current value that is the smallest between the two players (Bob or Candy). On the other hand, if player two (Bob) and player one (Candy) are close together, then Alice would choose a more distant location, by adding a 3 to the current value that is the largest between the two players (Bob or Candy), maximising her distance and boosting the possibility of a higher utility. This entire strategy can be considered as a **best response strategy** for Alice, a strategy that produces the most favorable outcome for a player, taking other players' strategies as given.

For example, if the candy's most frequent location is 5 o'clock and the bob's most frequent location is 6 o'clock. The clockwise and anticlockwise distances would be 1 and 11 respectively. According to this strategy, as the values are not 4 stops away but are close together and the anticlockwise direction is greater, thus a value of 3 is added to the largest value which in this case is Bob's location. Therefore, Alice's new location would be 9 o'clock which receives the highest utility of 11.

The aim of this game was to be able to develop different strategies for each of the players to maximise their utilities. My implementation of this game involved in designing three different strategies (one for each player), which ran simultaneously for 100

days. Each of these strategies had its own unique advantages and no strategy proved to be the best. The outcome of the strategies was the sum of the utilities for each of the players as shown in figure 3 which continued to change. These Strategies took both the cooperative and competitive environments into account and by creating these tactics, it strengthened the knowledge of sequential decision-making under uncertainty, non-cooperative and cooperative games, and agent-based modelling.



Sum of the utilies for player one (Candy):	898
Sum of the utilies for player two (Bob):	683
Sum of the utilies for player three (Alice):	819

Figure 3

Appendix

The Experimental Analysis:

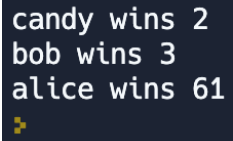
1. A terminal window with a dark background and light-colored text. It displays three lines of text: 'candy wins 2', 'bob wins 3', and 'alice wins 61'. Below the text, there is a small yellow cursor icon.

Figure 1

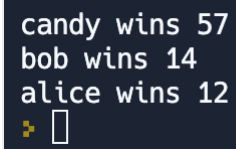
2. A terminal window with a dark background and light-colored text. It displays three lines of text: 'candy wins 57', 'bob wins 14', and 'alice wins 12'. Below the text, there is a small yellow cursor icon and a small white rectangular box.

Figure 2