Project 2

Title

Monopoly\*

\*A text-based computer replica of the

well-known classic boardgame –

for educational purposes only

Course

**CSC-5 Programming Concepts and Methodology I: C++**

Section

**40514**

Due Date

**June 7, 2020**

Author

**Sonya Smalley**

# Basic Gameplay

The object of this game is to be the only player left at the end who has not gone bankrupt. Players ‘move’ tokens around the virtual board according to the roll of the dice and take actions that vary according to the space that they land on. As in the original game, players collect $200 as they pass Go, and rolling doubles gives them an extra turn. However, if they get too lucky and roll three doubles in a row, they are sent to jail to reflect on the gravity of their crimes.

Due to time constraints, this game is more limited than the actual game of Monopoly (currently trademarked and published by Hasbro) in a few ways, which will be discussed in more detail throughout this writeup and which are listed in the Featured Not Included section of this writeup. Of these, the most notable features that are missing are the increased rent for monopolies and the ability to build up properties. However, due to the ability to use arrays, there have been some features added to the game which were not available at the completion of Project 1.

# Players

Monopoly can be played by 2 to 8 players. This is a major feature that was added to this game thanks to arrays. The user will be joined by anywhere from 1 to 7 other computer players.

# Player Actions

There are a limited number of actions that the player can take. Most actions are automatic or require only that they press the Enter key. For instance, money is automatically exchanged as appropriate, whether the player objects or not (rules are rules, and there are no deals or loans in *this* game). Some of the actions that require user input are instances when the player must make choices – such as choosing their game tokens, choosing Get Out of Jail options, and choosing whether or not to purchase available properties – and when it’s the player’s turn to roll. I had intended on adding the ability to build up properties by adding houses and hotels (a feature of the original game that allows players to charge higher rents); however, I ran out of time and was not able to add that feature. Had that feature been added, player interaction would have increased substantially. It would have required them to make decisions at the start of their turn and at the end regarding which actions they would like to take. Unfortunately, this would have required a lot of additional coding, for which there was just not enough time to complete (version 7 of this game, which was the final version required prior to adding said feature, was finished on the afternoon of June 6).

# Properties

In the original Monopoly game, players may buy and sell properties, mortgage them when additional funds are needed, build houses and hotels on them, obtain regional monopolies, and charge rents for owned properties that vary accordingly with each of the aforementioned conditions. Most of these abilities are not enabled in the game I have produced. However, in one of the early versions of this game (after Project 1 was submitted), the ability to purchase properties was added. The computer players will purchase the property if they have the funds, while the user has a choice to not do so. During the game, players will charge rent fees to other players who land on their owned properties. The value of each property is different and the rent to be charged varies according to location.

# Utilities

Utilities are a special type of property. Instead of paying a set rent that is based on how extensively properties are built out, the fees that players pay when they land on either Water Works or Electric Company are determined by the roll of the dice: a fee of $4 multiplied by the number shown after a roll of the dice. In the original game, the dice are multiplied by $10 if both utilities are owned by one player, although, this being a type of monopoly, that option was not included in my game.

# Go to Jail & Get Out of Jail

There are three ways to end up in jail: landing on the Go to Jail space, rolling three doubles in a row during one turn, and drawing a Chance or Community Chest card that tells you to Go to Jail. As soon as a player lands in jail, their turn is over. While they are in jail, they continue to collect rent fees as appropriate. (Note that just landing on the jail space does not put them in to jail, as they are just visiting in that case.)

There are similarly three ways that a player can get out of jail: pay $50 on their next turn, use a Get Out of Jail Free card, or wait for up to 3 turns. If they choose to pay $50 or use a Get Out of Jail Free card, they will roll the dice and move out of jail the number of spaces indicated on their dice. If they choose to wait for 3 turns and try to get out of jail without spending money or a Get Out of Jail Free card, they must roll the dice at each turn. If they get doubles, they are free from jail. They may then roll again and move their token as usual. However, if they do not get doubles by the end of their third turn, they are required to pay $50 to get free from jail and then continue their turn as usual (roll, then move).

Chance and Community Chest Cards

When players land on the Chance or Community Chest spaces, a random card is chosen for them. The player must then immediately follow the instructions given by the card. If a fee is charged or gifted to the player, then the requisite funds are automatically deducted from or added to their accounts. If they are instructed to move to a specific space, their game piece is immediately moved to the specified location. If they receive a Get Out of Jail Free card, they may keep the card for later use. I have added the restriction that players may only hold one Get Out of Jail Free card at a time. Therefore, if they already have one, the program will select from the cards again until a different card is chosen.

# Overview of Main Features

|  |  |
| --- | --- |
| **Feature** | **Programming Description** |
| 2 to 8 players | The number of players is randomly chosen and stored in a variable |
| Players may choose any of 13 tokens to play the game with | Looped input from a file, getline, and randomly chosen selection |
| Play goes in one direction around a board of 40 spaces | Each player’s movements are stored in an array and modified with const BOARD |
| Properties can be | Using a loop, randomly assigned ownership to 40 properties, output to a file |
| Rents, taxes, fees, etc. are automatically removed from or added to player accounts | Match player position value to loop input from the properties file created in a separate program, as well from a two-dimensional array in which costs associated with each property are stored |
| Play is facilitated by the simulated roll of the dice | Random number from 1 to 6 |
| Community Chest and Chance Cards | If statements determine if these spaces are landed on as well as what to do with each card; card info is read from a card file on loop |
| Determining who goes first and taking turns | Who goes first is determined by ‘rolling’ dice and comparing the values of the dice (highest goes first); turns are tracked with a variable and using an overarching do-while statement |
| Jail | A boolean variable with if statements to hold state of freedom; |
| Getting out of jail | A switch statement and a few variables |
| Doubles | A variable tracks the number obtained in a row; used in overarching while statement |

# Features Not Included

* Selling and mortgaging properties
* Adding houses and hotels to properties
* Rents that vary depending on monopolies and numbers of houses and hotels on properties
* Auctions
* Deals between players
* A quit-at-any-time option
* Realistic card deck usage (meaning that cards are discarded as they are used, then the deck is reshuffled once it is depleted)
* Players may have more than one Get Out of Jail Free card
* If a player chooses to use a Get Out of Jail Free card, the system verifies that they have one to use
* More player interaction

# Pseudocode

Global Constant

Game Board = 40 spaces

Max Players = 8

Function Prototypes

Set Random Number Seed

Declare Variables and Initialize

Choice – to hold user choices as necessary

Round – keep track of how many times round the board they go

Round Start – track which player’s turn marks the start of a new round

Turn – track whose turn it is

Number of Players – the number will be randomly chosen

Number of Bankrupt Players – track how many players have gone bankrupt

Property Costs – array to hold all the costs associated with each space on the board

Owner – array to indicate who owns each property

Players – array to hold the names of the tokens chosen by each player

User – variable to track which token is the user’s

Die1 and Die2 – two dice to be shared

Sumdie – sum the value of two rolled dice

Doubles – count of doubles per turn

Space – array to mark player positions on the board

Money – array to track player money

Bankrupt – array to track which players are bankrupt and out of the game

Jail – (Boolean) array to indicate if players are in jail or not

Jail Card – (Boolean) array: players may keep one Get Out of Jail Free card each

Jail Turns – array to keep track of how many turns in a row a player is in jail (max 3)

Property – holds the name of a property

End Game – (Boolean) indicates the end of the game if true

End Player – vector to track which round a player is out of the game

Position – holds a space number

Owned By – holds the owner of a property

**Display Game Piece User Menu**

Game pieces are chosen using the gamePieces.dat file

Ask Player 1 to choose their game piece token

Player 1 selects a number and chooses their game piece token

Loop to randomly assign a different game piece token to other players

Loop to pull Player’s choices from the file and then verify their choice

If statements to ensure only one player uses any token

**Roll to See Who Goes First** – Highest Value Goes First

Loop the whole process

Tell user what is going on and prompt them to start the process

Roll die1 for Player 1 and die2 for Player 2

Output results to user

If-else statements to see who rolled highest

Output who goes first

If they rolled the same value, then repeat the entire process again

**Loop to play the game**

Test to see if player is in jail

If in jail, do **Get Out of Jail activities**

Track how many turns in a row the player is in jail (max 3)

Prompt user to choose pay, card, or roll (randomize the choice for computer players)

Validate user input and loop if there’s an error

If player pays, money is deducted and they’re freed

If player uses a card, the card is no longer in their possession and they are freed

If player rolls for doubles, there are three possible outcomes

If they roll doubles, they are freed

If it is their first or second turn in jail and they do not roll doubles, then they must try again on their next turn

If is their third turn in jail and they do not roll doubles, they pay the fee, and then are freed

If not in jail, proceed with **Regular Game Play**

Play is on a loop as long as the player rolls doubles and is not in jail

Roll two dice

Sum the dice and move the token the number of spaces indicated by the sum

If the token will move off the board, go round the board instead

If they land on Go (space 1) or pass Go, output that information and add $200 to their account

**Loop to check which space the player landed on**

If they land on chance or community chest, player is prompted to choose a card

Open cards file and loop to randomly select one

Output the card to the user

Test value of the card and take appropriate action dependent on which card it is

Some cards award money or take it away

Some send the player to different spaces on the board

Some put the player in jail

Some are Get Out of Jail Free cards, to be kept for later

If they land on Income Tax or Luxury Tax, money is deducted from their account

If they land on an ownable property, check to see who owns the property

If they own the property, then output that information and move on

If another player owns the property, search Property Costs array to determine how much rent is owed

Deduct the required funds from their account and add them to the appropriate player’s account

If the property is unowned, the player has the option to purchase the property

If the player chooses to purchase the property, deduct the appropriate funds and note the change of ownership in the ownership array

If the player rolled doubles, they roll again

Count how many times in a row a player rolls doubles

If the player rolls three doubles in a row, they go to jail and their turn ends

Else, the loop starts at the top of regular game play

**End the game**

Output some stats

Create a thank you certificate