Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
simulateDiceRoll()	1	Generates a random number between a predetermined range (1, 6)	N/A	Number Between 1 and 6	Function Generates a 4	P
simulateDiceRoll()	2	Generates a random number between a predetermined range (1, 6)	N/A	Number Between 1 and 6	Function Generates a 2	P
simulateDiceRoll()	3	Generates a random number between a predetermined range (1, 6)	N/A	Number Between 1 and 6	Function Generates a 6	P
getPlayerOnTile()	1	This function gets the symbol of the current player on the current tile, and returns the symbol	getPlayerOnTile(1, 1, 1, 1, 1, 4)	A “#”, which represents multiple players on 1 tile.	“#”	P
getPlayerOnTile()	2	This function gets the symbol of the current player on the current tile, and returns the symbol	getPlayerOnTile(1, 1, 0, 0, 0, 1)	Returns “A”	“A”	P
getPlayerOnTile()	3	This function gets the symbol of the current player on the current tile, and returns the symbol	getPlayerOnTile(20, 1, 1, 20, 1, 4)	Returns “C” - Player 3 position	“C”	P
movePlayerPosition()	1	Moves Each Player to their new position	movePlayerPosition(1, 6, 50)	Returns “7”	Returns “7”	P
movePlayerPosition()	2	Moves Each Player to their new position	movePlayerPosition(49, 1, 50)	Returns “50”	Returns “50”	P

movePlayerPosition()	3	Moves Each Player to their new position	movePlayerPosition(49, 5, 50)	Returns “46”	Returns “46”	P
getRandomNumber()	1	Generates a random number within the specified range.	getRandomNumber(-10,10)	Returns 10	10	P
getRandomNumber()	2	Generates a random number within the specified range.	getRandomNumber(-100,100)	Returns 56	56	P
getRandomNumber()	3	Generates a random number within the specified range.	getRandomNumber(-1000,1000)	Returns 540	540	P
generateMathProblem(int min, int max	1	Generates and displays a random math problem based on specified range and evaluates the player's answer.	generateMathProblem(-10,10)	Prints ”Solve: 5 + 5” If Correct, returns 1 If Wrong, returns 0	Prints ”Solve: 5 + 5” If Correct, returns 1 If Wrong, returns 0	P
generateMathProblem(int min, int max	2	Generates and displays a random math problem based on specified range and evaluates the player's answer.	generateMathProblem(-100,100)	Prints ”Solve: 50 + 50” If Correct, returns 1 If Wrong, returns 0	Prints ”Solve: 50 + 50” If Correct, returns 1 If Wrong, returns 0	P
generateMathProblem(int min, int max	3	Generates and displays a random math problem based on specified range and evaluates the player's answer.	generateMathProblem(-1000,1000)	Prints ”Solve: 500 + 500” If Correct, returns 1 If Wrong, returns 0	Prints ”Solve: 500 + 500” If Correct, returns 1 If Wrong, returns 0	P
generateHarderMathProblem(int min, int max	1	Generates and displays a	generateHarderMathProblem(-1	Prints ”Solve: 5 + 5 - 10 ”	”Solve: 5 + 5 - 10 ”	P

		random math problem based on specified range and evaluates the player's answer. With 3 Numbers and 2 Operators	0,10)	If Correct, returns 1 If Wrong, returns 0	If Correct, returns 1 If Wrong, returns 0	
generateHarderMathProblem(int min, int max	2	Generates and displays a random math problem based on specified range and evaluates the player's answer. With 3 Numbers and 2 Operators	generateHarderMathProblem(-100,100)	Prints ”Solve: 50 + 50 - 100 ” If Correct, returns 1 If Wrong, returns 0	”Solve: 50 + 50 - 100 ” If Correct, returns 1 If Wrong, returns 0	P
generateHarderMathProblem(int min, int max	3	Generates and displays a random math problem based on specified range and evaluates the player's answer. With 3 Numbers and 2 Operators	generateHarderMathProblem(-1000,1000)	Prints ”Solve: 500 + 420 - 69 ” If Correct, returns 1 If Wrong, returns 0	”Solve: 500 + 420 - 69 ” If Correct, returns 1 If Wrong, returns 0	P
rollDiceAndMove(int currentPlayer, int *playerPosition, int winningPosition)	1	Simulates a player's dice roll, moves the player's position, and displays their new position.	rollDiceAndMove (2, 1, 50)	Player should move from Position [2] to updated position based off dice roll.	“Player 2 Rolled a 6 and moved from [1] to new Position [7]”	
rollDiceAndMove(int currentPlayer, int *playerPosition, int winningPosition)	2	Simulates a player's dice roll, moves the player's position, and displays their new position.	rollDiceAndMove (3, 5, 50)	Player should move from Position [2] to updated position based off dice roll.	“Player 3 Rolled a 4 and moved from [5] to new Position [9]”	
rollDiceAndMove(int currentPlayer, int *playerPosition, int	3	Simulates a player's dice roll, moves the	rollDiceAndMove (1, 2, 50)	Player should move from Position [2] to updated position based off dice	“Player 1 Rolled a 4 and moved	

winningPosition)		player's position, and displays their new position.		roll.	from [2] to new Position [6]”	
generateAndCheckMathProblem(int numMin, int numMax)	1	Checks whether a player answers a math question correctly	generateAndCheckMathProblem(-10,10)	Returns 0 if player answers it wrong, and 1 if its correct.	1 Player answers correctly	P
generateAndCheckMathProblem(int numMin, int numMax)	2	Checks whether a player answers a math question correctly	generateAndCheckMathProblem(-100,100)	Returns 0 if player answers it wrong, and 1 if its correct.	0 Player gets it wrong.	P
generateAndCheckMathProblem(int numMin, int numMax)	3	Checks whether a player answers a math question correctly	generateAndCheckMathProblem(-1000,1000)	Returns 0 if player answers it wrong, and 1 if its correct.	1 Player answers correctly	P
handlePlayerEjection(int *playerPosition, int *numPlayers, int currentPlayer, int *gameStatus, int *winningPlayer)	1	Ejects a player from the game if their position is less than 0.	handlePlayerEjection(0, 4, 1, 0, 0)	Sets playerPosition to -1 and decrements the number of players	Sets playerPosition to -1 and decrements the number of players	P
handlePlayerEjection(int *playerPosition, int *numPlayers, int currentPlayer, int *gameStatus, int *winningPlayer)	2	Ejects a player from the game if their position is less than 0.	handlePlayerEjection(2, 4, 1, 0, 0)	Does nothing as player is in a valid Position	Does nothing as player is in a valid Position	P
handlePlayerEjection(int *playerPosition, int *numPlayers, int currentPlayer, int *gameStatus, int *winningPlayer)	3	Ejects a player from the game if their position is less than 0.	handlePlayerEjection(-1, 4, 2, 0, 0)	Sets playerPosition to -1 and decrements the number of players	Sets playerPosition to -1 and decrements the number of players	P
handleIncorrectAnswer(int currentPlayer, int *playerPosition, int *numPlayers, int *gameStatus, int *winningPlayer)	1	Handles the case when the player answers incorrectly, moving them back a random number of tiles.	handleIncorrectAnswer(1, 4 ,4, 0, 0)	If Player Gets it correct he stays in the tile, if he gets it wrong he moves back 1-3 Tiles.	He gets it correct and stays on [4]	P

handleIncorrectAnswer(int currentPlayer, int *playerPosition, int *numPlayers, int *gameStatus, int *winningPlayer)	2	Handles the case when the player answers incorrectly, moving them back a random number of tiles.	handleIncorrectAnswer(3, 2, 4, 0, 0)	If Player Gets it correct he stays in the tile, if he gets it wrong he moves back 1-3 Tiles.	He gets it wrong, moves back 3 tiles to - 1.	P
handleIncorrectAnswer(int currentPlayer, int *playerPosition, int *numPlayers, int *gameStatus, int *winningPlayer)	3	Handles the case when the player answers incorrectly, moving them back a random number of tiles.	handleIncorrectAnswer(1, 50, 2, 0, 0)	If Player Gets it correct he stays in the tile, if he gets it wrong he moves back 1-3 Tiles.	He gets it right and stays in the same Tile [50]	P
playTurn()	1	Plays a single turn for the current player, handling dice roll, movement, and math problem interaction.	gameStatus=0, currentPlayer=1, playerPosition=49, winningPosition=50, dice roll = 1	gameStatus=1, winningPlayer=1.	gameStatus=1, winningPlayer=1.	P
playTurn()	2	Plays a single turn for the current player, handling dice roll, movement, and math problem interaction.	currentPlayer=2, answerCorrect=0, playerPosition=3, numPlayers=3	Prints "Player 2 ejected.", playerPosition=-1, numPlayers=2.	Prints "Player 2 ejected.", playerPosition=-1, numPlayers=2.	P
playTurn()	3	Plays a single turn for the current player, handling dice roll, movement, and math problem interaction.	currentPlayer=3, dice roll = 6	Prints "Player Rolled a 6!: Extra turn for Player [3]", gameplay continues with another dice roll.	Prints "Player Rolled a 6!: Extra turn for Player [3]", gameplay continues with another dice roll.	P