	Function	#	Description	Sample input Data	Expected Output	Actual	P/F
	checkTile(scouter: Animal)	1	Move Player 2 Leopard to unoccupied river tile (row 5, column 5)	w	validity = 0; "Sorry, you can't move here!"	validity = 0; "Sorry, you can't move here!"	P
	checkTile(scouter: Animal)	2	Move Player 2 Rat to unoccupied river tile (row 5, column 5)	d	validity = 1; "Tile verified! Scouting for enemies..."	validity = 1; "Tile verified! Scouting for enemies..."	P
	checkTile(scouter: Animal)	3	Move Player 2 Leopard to unoccupied trap tile (row 3, column 1)	a	validity = 1; selectedAnimal.strLv = -1; "Trapped! Your Leopard has been weakened"	validity = 1; selectedAnimal.strLv = -1; "Trapped! Your Leopard has been weaken	P
	checkTile(scouter: Animal)	4	Move Player 2 Rat to unoccupied trap tile (row 3, column 1)	a	validity = 1; selectedAnimal.strLv = -1; "Trapped! Your Rat has been weakened"	validity = 1; selectedAnimal.strLv = -1; "Trapped! Your Rat has been weakened"	P
	checkTile(scouter: Animal)	5	Move Player 2 Leopard to Upper unoccupied grass tile (row 5, column 6)	w	validity = 1; "Tile verified! Scouting for enemies..."	validity = 1; "Tile verified! Scouting for enemies..."	P
	checkTile(scouter: Animal)	6	Move Player 1 Rat to Player 2 Homebase	d	"You made it to the enemy base!"; players[0].setWinstate = 1;	"You made it to the enemy base!"; players[0].setWinstate = 1;	P
	checkTile(scouter: Animal)	7	Move Player 2 Rat to Player 1 Homebase	a	"You made it to the enemy base!"; players[1].setWinstate = 1;	"You made it to the enemy base!"; players[1].setWinstate = 1;	P
	checkTile(scouter: Animal)	8	Move Player 1 Rat to Player 1 Homebase	a	"That's your own base, genius."	"That's your own base, genius."	P
	checkTile(scouter: Animal)	9	Move Player 2 Rat to Player 2 Homebase	a	"That's your own base, genius."	"That's your own base, genius."	P
	scoutTile(scouter: Animal)	1	Player 2 Leopard: scout an unoccupied grass tile	a	occupation = 'o'	occupation = 'o'	P
	scoutTile(scouter: Animal)	2	Player 2 Leopard: scout an ally-occupied grass tile	s	occupation = 'b'	occupation = 'b'	P
	scoutTile(scouter: Animal)	3	Player 2 Rat: scout an enemy-occupied grass tile (equal enemy)	d	occupation = 'e'	occupation = 'e'	P
	scoutTile(scouter: Animal)	4	Player 2 Rat: scout an enemy-occupied grass tile (stronger enemy)	d	occupation = 'e'	occupation = 'e'	P
	scoutTile(scouter: Animal)	5	Player 2 Rat: scout an enemy-occupied river tile	d	occupation = 'b'	occupation = 'b'	P
	battle()	1	Player 2 Leopard attacks Player 1 Rat	w	battleWinner = 2; "Player 1's Rat has been defeated!"	battleWinner = 2; "Player 1's Rat has been defeated!"	P
	battle()	2	Player 2 Rat attacks Player 1 Rat	s	battleWinner = 2; "Player 1's Rat has been defeated!"	battleWinner = 2; "Player 1's Rat has been defeated!"	P
	battle()	3	Player 2 Rat attacks Player 1 Leopard	a	battleWinner = 1; "Your animal has died."	battleWinner = 1; "Your animal has died."	P
	battle()	4	Player 2 Leopard attacks Player 1 Leopard	d	battleWinner = 2; "Player 1's Leopard has been defeated!"	battleWinner = 2; "Player 1's Leopard has been defeated!"	P
	battle()	5	Player 2 Leopard attacks Player 1 Leopard (trapped)	d	battleWinner = 2; "Player 1's Leopard has been defeated!"	battleWinner = 2; "Player 1's Leopard has been defeated!"	P
	battle()	6	Player 2 Rat attacks Player 1 Leopard (trapped)	a	battleWinner = 2; "Player 1's Leopard has been defeated!"	battleWinner = 2; "Player 1's Leopard has been defeated!"	P
	battle()	7	Player 2 Rat attacks Player 1 Rat (trapped)	w	battleWinner = 2; "Player 1's Rat has been defeated!"	battleWinner = 2; "Player 1's Rat has been defeated!"	P
	battle()	8	Player 2 Rat attacks Player 1 Elephant	a	battleWinner = 2; "Player 1's Elephant has been defeated!"	battleWinner = 2; "Player 1's Elephant has been defeated!"	P
	battle()	9	Player 2 Lion leaves trap to attack Player 2 Wolf	w	battleWinner = 2; "Player 1's Wolf has been defeated!"	battleWinner = 2; "Player 1's Wolf has been defeated!"	P
	GameModel()	1	MVC: Game model instantiation	n/a	Creates a model of Jungle King and all associated objects and classes	Creates a model of Jungle King and all associated objects and classes	P
	GameView()	1	MVC: Game view instantiation; visual representation of the board, pieces, and tiles	n/a	Creates a view to communicate the Jungle King model and its components to user	Creates a view to communicate the Jungle King model and its components to user	P
	GameController()	1	MVC: Handles user inputs and allows them to communicate with the model and view	n/a	Allows the user to manipulate the model and update views	Allows the user to manipulate the model and update views	P
	GameLauncher()	1	A class that instantiates the MVC architecture in one platform	n/a	Instantiates the MVC architecture of Jungle King in one platform	Instantiates the MVC architecture of Jungle King in one platform	P
	controllerSelect()	1	Select the Leopard to be the active animal (animal index: 3)	3	"You have selected Leopard."	"You have selected Leopard."	P
	controllerSelect()	2	Select the Rat to be the active animal (animal index: 4)	4	"You have selected Rat."	"You have selected Rat."	P
	controllerSelect()	3	Select animal index 9 to be the active animal	9	"Invalid input. Try again."	"Invalid input. Try again."	P
	controllerSelect()	4	Select the Rat when Rat is dead	4	Rat is not available for select	Rat is not available for select	P
	controllerMove()	1	Move Player 2 Leopard to Right unoccupied grass tile (row 6, column 7)	d	tiles[6][7].status = 'l'	tiles[6][7].status = 'l'	P
	controllerMove()	2	Move Player 2 Leopard to Upper unoccupied grass tile (row 5, column 6)	w	tiles[5][6].status = 'l'	tiles[5][6].status = 'l'	P
	controllerMove()	3	Move Player 2 Leopard to Lower OOB tile (row 7, column 6)	s	"Let's try that one again! Returning to animal select..."	"Let's try that one again! Returning to animal select..."	P
	controllerMove()	4	Move Player 2 Leopard to Left unoccupied grass tile (row 6, column 5)	a	tiles[6][5].status = 'l'	tiles[6][5].status = 'l'	P
	controllerMove()	5	Move Player 2 Leopard to unoccupied river tile (row 5, column 5)	w	tiles[5][5].status = 'R'; "Doesn't look like Leopard knows how to swim!"	tiles[5][5].status = 'R'; "Doesn't look like Leopard knows how to swim!"	P
	controllerMove()	6	Move Player 2 Rat to unoccupied river tile (row 5, column 5)	d	tiles[5][5].status = 'r'; "Looks like Rat can swim after all!"	tiles[5][5].status = 'r'; "Looks like Rat can swim after all!"	P
	controllerMove()	7	Move Player 1 Lion to unoccupied river tile (row 6, column 4)	d	Lion moved to row 6, column 7	Lion moved to row 6, column 7	P
	controllerMove()	8	Move Player 1 Lion to unoccupied river tile (row 6, column 4), but col 7 is obstructed	d	Lion does not move	Lion does not move	P
	controllerMove()	9	Move Player 2 Lion to unoccupied river tile (row 3, column 6), but row 0 col 6 is obstructed	w	Lion does not move	Lion does not move	P
	controllerMove()	10	Move Player 2 Lion to occupied river tile (row 3, column 6)	w	Lion does not move	Lion does not move	P
	keyPressed()	1	Player 1 inputs 1 and then w (game start)	1 -> w	Display message: "Out of Bounds"	Display message: "Out of Bounds"	P
	keyPressed()	2	Player 1 inputs 1 and then d	1 -> d	Elephant moves to the right	Elephant moves to the right	P
	keyPressed()	3	Player 1 inputs 6 and then a	6 -> a	Dog moves left	Dog moves left	P
	keyPressed()	4	Player 1 inputs 6 and then s	6 -> s	Dog moves down	Dog moves down	P
	keyPressed()	5	Player 1 inputs 6 and then w	6 -> w	Dog moves up	Dog moves up	P
	keyPressed()	6	Player 1 inputs 6 and then d	6 -> d	Dog moves right	Dog moves right	P
	keyPressed()	7	Player 1 inputs 9	9	Player 1 reprompted to select	Player 1 reprompted to select	P
	keyPressed()	8	Player 1 inputs 1 and then x	1 -> x	Player 1 reprompted to select	Player 1 reprompted to select	P
	keyPressed()	9	Player 1 inputs 1 when p1 elephant is captured	1	Player 1 reprompted to select	Player 1 reprompted to select	P
	keyPressed()	10	Player 1 inputs 1 when no animals are available	1	Player 1 turn ends	Player 1 turn ends	P
	keyPressed()	11	Player 2 captures home base	8 -> a	Game over; Display: "P2 wins!"	Game over; Display: "P2 wins!"	P
	Coinflip() -- edit	1	Player1 inputs -99	-99	Invalid choice. Please pick a valid number	Invalid choice. Please pick a valid number	P
	Coinflip() -- edit	2	Player1 inputs 2, Player2 inputs 2	2, 2	Invalid choice. Please pick a valid number(P2)	Invalid choice. Please pick a valid number	P
	Coinflip() -- edit	3	Player1 inputs 2, Player2 inputs 3	2, 3	Player1 or Player2 have a higher strenght and game continues	Player1 or Player2 have a higher strenght and game continues	P
	Coinflip() -- edit	4	Player inputs Hello	Hello	Invalid input. Please enter a number	Invalid input. Please enter a number	P