Test	Description	Inputs	Expected Outputs	Actual Outputs	Test Creation Date	Pass/ Fail	Outcome Description
Property Test							
1	Creation of a property object with the description, price and property level	Description: Robbie's Home Price: 250 Rents: 20, 40, 60, 80, 100, 120	Getter methods to return correct description, price and correct rent for each property level	The expected outputs	06/05/2018	PASS	Property objects are properly created
2	Should receive an error when property is created with invalid parameters	Description: null Price: -250 Rents: null	Illegal argument exception	Illegal argument exception	06/05/2018	PASS	Error correctly given for improper parameters
3	Creation of a station with a description and price	Description: Matt's Rail Price: 200	Getter methods to return correct description and price for the station	The expected outputs	06/05/2018	PASS	Stations are properly created

4	Should receive an error when station is created with invalid parameters	Description: null Price: -200	Illegal argument exception	Illegal argument exception	06/05/2018	PASS	Error correctly given for improper parameters
5	Creation of a utility with a description and price	Description: Ollie's Electrics	Getter methods to return correct description and price for the utility	The expected outputs	06/05/2018	PASS	Utility was properly created
6	Should receive error when utility is created with invalid parameters	Description: null Price: -200	Illegal argument exception	Illegal argument exception	06/05/2018	PASS	Error correctly given for improper parameters
7	Property objects that are bought should be owned by the correct player	Player object: buyer=player(jim,coral) Call buy(buyer) from Property class	getOwner() in property should return buyer object and isOwned() should be true	Buyer object is returned and isOwned() is true	06/05/2018	PASS	Correct player is given ownership of the property

8	Should receive an error when a null buyer object tries to buy a property	Player object: buyer=null Call buy(buyer) from Property class	getOwner() in Property should return an error and instance isOwned() returns null	The expected results	06/05 /2018	PASS	Error correctly given for improper parameters

9	Downgrade() should return the cash amount that should be credited for and a property's level should go down	cash=instance.downgrade();
		instance.getImprovementCost();
		
		getLevel(0);
			Cash should equal getImprovementCost()
			Instance.getLevel() should be 0
				The expected results	06/05 /2018	PASS	The correct amount is returned and the property level is decremented unless the min level

10	House properties should be part of a group	instance = Property.create("T 's Gym", 500, new int[]{1,1,1,1,1,1}); Instance should be assigned to a new group 'group' using property.group.create method.	Instance.isGrouped() to return true and group equal instance.getGroup()	The expected results	06/05 /2018	PASS	The property was assigned to the correct group

11	Should be able to get cost of upgrading or downgrading a property	improve = instance.getImprovementCost(); where instance is a created property and it assigned to a group	Improve should be equal to the groups getImprovementCost method	The expected results	06/05/2018	PASS	The correct amounts are returned
12	Should be able to get amount you would receive for mortgaging a property	mortgaged = instance.getMortgagedPrice() where instance is a property	Mortgaged should be equal to half the amount returned from instance.getPrice()	The expected results	06/05/2018	PASS	Mortgaging works as it should
13	Should be able to get the rent price of a property	Property called instance, which is assigned to a group ‘group’, and then bought by a player and later upgraded	instance.getRentPrice(group.getLevels().getLevel(0) should return 1 and after instance.upgrade() is called, 5	The expected result	06/05/2018	PASS	Rent price returned works when a house is put on the property

14	Should be able to get the rent price of a station	Property called instance which is assigned to a group ‘group’ and then bought by a player. After the same player buys another station.	station1.getRentPrice(group.getLevels().getLevel(0) should return 25 and station2 should return 50	The expected result	06/05/2018	PASS	Works
15	Should be able to get rent price of utility, depending on dice throw and number of utilities owned by player	Two property objects utility1 and utility2, where they’re both assigned to a group and owned by the same player	Rent price should be 24 when one utility is owned and the dice throw equals 6 (4 times dice throw). Rent price should be 70 when two utilities are owned and dice throw equals 10 (10 times dice throw).	The expected result	06/05/2018	PASS	Works
16	A player should be able to sell a property it bought for the original amount	Property ‘instance’ where it’s owned by Player ‘buyer’ and then sold sellPrice equals instance.sell()	sellPrice should equal the getPrice method and the property shouldn’t be owned anymore	The expected result	06/05/2018	PASS	The correct amount is returned

17	Should be able to upgrade a property level of a property	Property ‘instance’ where its assigned to a group, bought by Player ‘buyer’ and upgraded	Instance.getLevel() should equal 1	The expected results	06/05 /2018	PASS	Level increased from 0 to 1
18	Should be able to set a property to a group	Property ‘instance’ which is assign to Group ‘group’	Instance.getGroup() should equal group	The expected results	06/05 /2018	PASS	Getter returns the correct group
19	Should be able to get the current property level of a property or utility	Property ‘instance’ and ‘utility’ where they’re assigned to different groups	getLevel() call for both instance and utility should both equal 0	The expected results	06/05 /2018	PASS	Works
20	Should be able to get the owner of a property	Property ‘instance’ which calls its buy method on a Player ‘buyer’	Buyer should be equal to instance.getOwner() and isOwened() should be true	The expected results	07/05 /2018	PASS	Property is owned by the specified buyer

21	Should be able to get the price of a property	Property ‘instance’ which is initialised with a cost of 50	The getPrice() method should return 50 and isMortgaged() should be false	The expected results	07/05 /2018	PASS	Returned 50
22	Should be able to check whether a property, house or utility is grouped	Property ‘instance’, ‘utility’ and ‘station’ are all created and added to a different group	isGrouped() should be true for all Property objects	The expected results	07/05 /2018	PASS	Grouping properties work
23	Should be able to check whether a property, utility and station is improvable(i.e . add houses)	Property ‘instance’, ‘utility’ and ‘station’ are all created and added to a different group	isImprovable() should be true for all Property objects	The expected results	07/05 /2018	PASS	All newly created properties are improvabl e
24	Should be able to check whether a property is mortgaged	Property ‘instance’ is created, added to a group, bought by Player ‘buyer’ and then calls mortgage()	Ismortgaged() should be true after mortgage() has been called	The expected results	07/05 /2018	PASS	Mortgagin g system works

25	Should be able to check whether a property is owned or not, and it should change when bought/sold	Property ‘instance’ which is bought and then sold Player ‘buyer’ is used to buy instance	isOwned() should be false before instance.buy(buyer) if called, true after , and false when instance.sell() is called	The expected results	07/05 /2018	PASS	Buying/selling system works
26	A player should not be able to interact with a property it has sold	Property ‘instance’ where it has been bought by a Player ‘buyer’ and later sold	Instance.isValid() should be true once instance has been bought, and false once it has been sold	The expected results	07/05 /2018	PASS	Properties aren’t valid once sold by player
27	Should be able to mortgage a property once it has been bought	Property ‘instance’ where it has been bought by Player ‘buyer’ and has called mortgage()	getMortgagedPrice() should equal half getPrice() and isMortgaged() should equal true	The expected results	07/05 /2018	PASS	The correct amount was returned
28	Should be able to un-mortgage a property once it has been mortgaged	Property ‘instance’ where its bought by Player ‘owner’, mortgaged and then un-mortgaged	Instance.getPrice() should equal 50 (given as cost parameter for instance), and isMortgaged() should return false	The expected results	07/05 /2018	PASS	The correct amount was returned

Property Level Group Test							
29	Test creation of PropertyLevel	Group ‘group’ where two PropertyLevel objects are used as parameters in its create method	Group should be improvable (true)	The expected results	07/05 /2018	PASS	
30	A group should be able hold PropertyLevel objects	Group ‘group’ where its create method takes in PropertyLevel ‘level1’ and ‘level2’	Contains(level2) should be true	The expected results	07/05 /2018	PASS	
31	Should be able to get the index of a group level	Int level2Index which equals group.getIndex(level2).	Level2Index should equal once as two PropertyLevel objects were added to the group	The expected results	08/05 /2018	PASS	
32	Should be able to get the level count of the group	Group ‘group’ where its create method took in two PropertyLevel objects in as parameters	getLevelCount() should return 2	The expected results	08/05 /2018	PASS	

33	Should be able to retrieve the highest level in the group(most houses)	PropertyLevel ‘Level2’ where two PropertyLevel objects have been added to a group	The getMax() method should return level2	The expected results	08/05 /2018	PASS	
34	Should be able to retrieve the minimum level in the group (least houses)	PropertyLevel ‘level1’ where two PropertyLevel objects have been added to a group	The getMix() method should return level1	The expected results	08/05 /2018	PASS	
35	Should be able to get next highest level based on a given level	PropertyLevel ‘nextLvl’ where two PropertyLevel objects have been added to a group and the first is used in the getNext() method	nextLvl should equal level2	The expected results	08/05 /2018	PASS	
36	Should be able to get the previous level of a given level	PropertyLevel ‘prevLvl’ where two PropertyLevel objects have been added to a group and the second is used in the	prevLvl should be equal level1	The Expected results	08/05 /2018	PASS	The previous level was returned.

		getPrevious() method					
37	Should be able to check whether the property is improvable or not	Two propertyLevel objects added to a group	Should return true as improvable as there are two property levels in the group .	The expected results	08/05 /2018	PASS	The isImprovable method works correctly
38	Should be able to check the maximum level of the group	Two propertyLevel objects added to a group	Should return the second property level object that was added into the group	The expected results	08/05 /2018	PASS	The isMax method works correctly
39	Should be able to check the minimum level of the group	Two propertyLevel objects added to a group	Should return the first property level object that was added into the group	The expected results	08/05 /2018	PASS	The isMin method works correctly

Property Group Test

40	Should be able to create different groups of properties with a particular description and colour	Three different property objects of regular property, station and utility added into their respective groups	Should return the correct groups. Should return the correct group descriptions. Should return the correct group colours.	The expected results	08/05 /2018	PASS	The create method in property group works correctly
41	Should be able to check if the correct group colour gets returned	One property object created and added into a group with a particular colour	Should return the correct group colours.	The expected results	08/05 /2018	PASS	The get colour method works correctly
42	Should be able to check if the correct group description get returned	One property object created and added into a group with a particular description	Should return the correct group description.	The expected results	08/05 /2018	PASS	The getDescri ption method works correctly
43	Should be able to get the property with the most houses in a group	Two property objects created and added into a group. A Player object created as the properties need to	Should return the first property object that was added into the group as that property has been upgraded the most.	The expected results	08/05 /2018	PASS	The getHighes tLevel method works correctly.

		be owned to be able to upgrade. First property upgraded three times and the second only twice.					
44	Should be able to get the property with the least amount of houses in a group	Two property objects created and added into a group. A Player object created as the properties need to be owned to be able to upgrade. First property upgraded three times and the second only twice.	Should return the second property object that was added into the group as that property has been upgraded the least.	The expected results	08/05 /2018	PASS	The getLowestLevel method works correctly.
45	Should be able to check the improvement cost of the properties in a group	Two property objects created and added into a group. The group has a fixed figure for improvement cost	Should be able to get the correct amount for improvement cost in a group	The expected results	08/05 /2018	PASS	The getImprovementCost works correctly.

46	Should be able to check whether a group is improvable or not	One regular property and one utility property object created and added in their respective groups.	Should return that the regular property group is improvable. Should return that the utility property group is not improvable.	The expected results	08/05 /2018	PASS	The isImprovable method works correctly
47	Should be able to check what type of levels a group contains (regular,station or utility)	Two property objects added into a group. The group has Level type of Regular	Should return Regular as the levels for the group	The expected results	08/05 /2018	PASS	The getLevels method works correctly
48	Should be able to check the owner of a particular group	Two property objects added into a group. A player object created to be able to buy all properties in a group and own the group	Should return the player that owns the group	The expected results	08/05 /2018	PASS	The getOwner method works correctly
49	Should be able to check what properties are contained in a group	Two property objects added into a group.	Should return true if the group contains the given properties	The expected results	08/05/ 2018	PASS	The getProperties method works correctly

50	Should be able to check whether a group is owned or not	Two property objects added into a group. A player object created to be able to buy all properties in a group and own the group	Should return true, showing that the group is owned.	The expected results	08/05/ 2018	PASS	The isOwned method works correctly	
Card Test								
51	Should be able to check if a card with multiple actions is created correctly	A card object is created which has multiple actions	Should return the correct description for the card. Should return whether the card has to be used immediately or not. Should return a particular action	Expected results	08/05/ 2018	PASS	The create method with 3 arguments works correctly	
52	Should be able to check if a card with one action is created correctly	A card object is created which has a cardAction object passed into it	Should return true if the description of the card is equal to the description of the cardAction	Expected results	08/05/ 2018	PASS	The create_card Action_boole an works correctly	

53	Should be able to get a the number of action in a card	A card object is created with multiple actions	Should return the correct number of actions in the card	Expected results	08/05/2018	PASS	The getActionCount method works correctly	
54	Should be able to get the description of a card with one action	A card object is created which has a cardAction passed into its description parameter	Should return true if the cardAction is equal to the description of the Card object	Expected results	08/05/2018	PASS	The getActionDescription method works correctly	
55	Should be able to get the description of a card with multiple description	A card object created with a particular description	Should return true if description is equal to the description of the card	Expected results	08/05/2018	PASS	The getDescription method works correctly	
56	Should be able to check what group a particular card belongs to	Two card objects are created and added to different groups.	Should return true if the expected groups is equal to the actual groups	Expected results	08/05/2018	PASS	The getGroup method works corrcetly	

57	Should be able to check whether a card is in a group	One card object created that is added into a group	Should return true as the card is in a group	Expected results	08/05/2018	PASS	The isGrouped method works correctly	
58	Should be able to check if the card gives you multiple choices	Two card objects created, one with only one action and the other with multiple	Should return true for the card with multiple actions. Should return false with the card with only one action.	Expected results	08/05/2018	PASS	The isChoice method works correctly	
59	Should be able to check whether the card has to be used immediately	One card object created with the immediate parameter set to true.	Should return true as the immediate parameter was set to true	Expected results	08/05/2018	PASS	The isImmediate method works correctly	
60	Should be able to check if atleast one action associated with the card is usable	One card object created with multiple actions	Should return true if any of the actions are usable.	Test failed as method was not implemented	08/05/2018	FAIL	The isUsable method with 0 arguments has not been implemented	

61	Should be able to check whether a card with multiple actions is usable	One card object created with multiple actions	Should return true if a chosen action is usable	Expected results	08/05/2018	PASS	The isUseable method with an argument works correctly	
62	Should be able to check whether a card is valid or not (valid if a player can use it)	One card object created that is added into a group. A player object has been created so that they can draw that particular card	Should return true as the player has drawn the card and is valid to use	Expected results	08/05/2018	PASS	The isValid method works correctly	
63	Should be able to check whether a player can use a card with one action	A card object is created that only has one action and is added into a group. A player object is created so that the card can be drawn and then used.	The isValid method should return true when the card has been drawn but should return false when the use method has been called on the card.	Expected Results	08/05/2018	PASS	The use method with 0 arguments works correctly	

64	Should be able to check whether a player can use a card with multiple actions	A card object is created that has multiple actions and is added into a group. A player object is created so that the card can be drawn and then used.	The isValid method should return true when the card has been drawn but should return false when the use method with an int representing the action of choice has been called on the card.	Expected results	08/05/2018	PASS	The use method with arguments works correctly	
65	Should be able to check whether a player owns a card	A card object is created and then is added into a group. A player object is created so that the card can be drawn and then used.	The is owned method should return false first as the card is not drawn. It then should return true when the card is drawn. After the card is used, the isOwned method should return false.	Expected results	08/05/2018	PASS	The isOwned method works correctly.	
66	Should be able to get the owner of a particular card	A card object is created and then added into a group. A player object is created so that the card can be owned.	The method will throw an error if the card hasn't been drawn. Once the card has been draw by a player the, the method should return that particular player as the owner.	Expected results	08/05/2018	PASS	The getOwner method works correctly.	

Card Group Test								
67	Should be able to check if a card group has been created.	Two card objects have been created and then added into different groups which have different descriptions for their parameter.	Should return true if the description of the group is equal to the set description.	Expected results	08/05/2018	PASS	The create method works correctly	
68	Should be able to check if a card can be drawn from the group	A card object has been created and then added into a group. A player object has also been created so that a card can be drawn from the group.	When the draw method is called, the card that has been drawn should have an owner. This would show that the method works correctly.	Expected results	08/05/2018	PASS	The draw method works correctly	

69	Should be able to check if a card can be replaced to the group	A card object has been created and the added to a group. A player object has also been created so that the the card in the group can be drawn, then used and then replaced.	When the card is drawn, there will be an owner for the card. When the card has been used and when the replace method has been called, the owner of the card should be null, indicating that the card has been replaced in the group.	The method has not been implemented so no error was thrown.	08/05/2018	FAILED	The replace method works correctly	

Property Level Test							
70	PropertyLevel objects should be ordered in level	PropertyLevel ‘level1’ and ‘level2’ where they both are in the same group	The compareTo() method should return -1 when level1 is compared to level2, 0 when level1 is compared to itself, and 1 when level2 is compared to level1	The expected results	08/05/2018	PASS	Property levels increase as level does

71	Should be able to get description of a PropertyLevel object	String “One House” , which is used as a parameter when PropertyLevel ‘level1’ is initialised	getDescription() should return One House	The expected results	08/05/2018	PASS	Works
72	Should be able the group to which a PropertyLevel object belongs to	Group ‘group’ where its create() method takes in 2 PropertyLevel ‘level1’ and ‘level2’.	getGroup() should return group	The expected results	08/05/2018	PASS	Correct group is returned every time
73	Should be able to get the index of a PropertyLevel obect in a group	Level2Index, where int level2Index = group.getIndex(level2)	Index of level2 should be 1	The expected results	08/05/2018	PASS	Index system works correctly
74	Should be able to check whether a PropertyLevel object part of a group	PropertyLevel ‘Level2’, where its used as a parameter the create() method in Group	Should return true	The expected results	08/05/2018	PASS	Works

75	Should be able to check if a PropertyLevel object is the highest in a group	PropertyLevel ‘level1’ and ‘level5’ where they’re both in the same group	IsMax() should return false for level1 and true for level5	The expected results	08/05/2018	PASS	Works
76	Should be able to check if a PropertyLevel object is the lowest in a group	PropertyLevel ‘level1’ and ‘level5’ where they’re both in the same group	IsMin() should return true for level1 and false for level2	The expected results	08/05/2018	PASS	Works
77	Should be able to get the next highest property level given a PropertyLevel object	PropertyLevel ‘nextIn , which holds the object that group.getNext(level1) returns	nextIn should be equal to level5	The expected results	08/05/2018	PASS	Highest level property always returned

78	Should be able to get the next lowest property level given a PropertyLevel object	PropertyLevel ‘before , which should hold the object that group.getPrevious(level5) returns	Before should be equal to level1	The expected results	08/05/2018	PASS	Lowest level property always returned
Player Test							
79	Player should be able to buy a Property object and it should cause their cash to decrement to the cost of the property	Player ‘player’, int ‘startCash’ and Property ‘instance’	Before player buys instance, instance.getOwner should be null, and equal player after. Also after purchase player.getCash() should return the amount it had before minus the cost of instance	The expected result	08/05/2018	PASS	Buy method properly deducts cash and affects a property

80	Player should be able to sell a house and get the money back that it previously spent to get it	Player ‘player’ and int ‘cash’. Property ‘instance’ is what the player will buy and initially add a house to	Once upgrade(instance) is called by player, players cash should equal what it was minus the improvement cost of instance. Once downgraded the players cash should equal what it was before taking away the improvement cost	The expected result	08/05/2018	PASS	sell method properly deducts cash and affects a property
81	A cards owner should be equal to the player that drew it	Player ‘player’, which calls draw(group) and Card ‘card’	Player should be equal to card.getOwner()	The expected result	08/05/2018	PASS	Card ownership works
82	Should be able to get the amount of cash a player has	1500, which is the default amount a Player has	1500 should equal player.getCash()	The expected result	08/05/2018	PASS	Cash getter works

83	Should be able to get colour associated with player	Color.Coral, which is what Player ‘player’ is instantiated with	Player.getColour should equal coral		08/05/2018	PASS	Colour getter works
84	Should be able to get description (name) of a player	String ‘T’, which is what Player ‘player’ is instantiated with	Player.getDescription() should equal	The expected result	08/05/2018	PASS	Description getter works
85	Player should be able to mortgage a property it has bought and receive half the amount it paid for it	Player ‘player’ and int ‘cash’, where player buys Property ‘instance’ and mortgages it.	Cash should equal the players new cash amount by using getMortgagePrice() and taking it away from its getPrice()	The expected result	08/05/2018	PASS	Mortgaging system changes players cash correctly

86	Player should have to pay another player a Property's rent if it lands on a property it owns	Int 'p1cash','p2cash' and Player 'player1' and 'player2'.	payRent(instance,5) should change both players cash so that instance gains the amount specified in position 5 (hotel rent). p1Cash should always equal player1.getCash(), and the same logic for p2Cash.	The expecte d result	08/05/2018	PAS S	Renting system works well using payRent method
87	Player should be able to sell a property it bought and gain back the money it spent on it	Int 'p1Cash' and Player 'player', where p1Cash initially equals the amount it had before the sale, and after equal the amount plus the cost of the property it sold	P1Cash should equal player.getCash() , as the sell() method should update player cash	The expecte d result	08/05/2018	PAS S	Works

88	Player should be able to unmortgage a property it previously mortgaged and pay half the amount of the property	Int ‘p1Cash’ and Player ‘player’ where p1Cash should equal the amount player has after unmortgaging Property ‘instance’	P1Cash and player.getCash() should be the same	The expected result	08/05/2018	PASS	Works
89	Player should be able to upgrade a property (add a house to it), and the improvement cost should deduct from players cash	Player ‘player’ and int ‘cash’ , where cash should equal the players cash after it upgrades a property using upgrade()	cash should equal player.getCash() , as upgrade() should update player cash	The expected result	08/05/2018	PASS	Houses and hotels are correctly applied
90	Once a player uses a pot luck card, they shouldn’t own it anymore	Card ‘card’, which Player ‘player’ uses its use() method on	Once the use() method is called on card, card.isOwned() should be false	The expected result	08/05/2018	PASS	Pot luck cards aren’t owned once used