Unit Testing

This document was intended to track all unit test cases carried out on our code.

Incomplete Test	A test which isn't implemented correctly
Passed Test	The code currently passses this test
Failed Test	A failed test needs to be addressed
Redundant Test	A test for a removed peice of code

Test ID	Test Case	Context	Test Description	Test Data	Expected Result	Actual Result	Notes
1.01	Retreive aircract position	Aircraft.position	Testing the get function for the position attribute	testAircraft	[0, 0, 0]	[0, 0, 0]	
1.02	Retreive aircraft name	Aircraft.name	Testing the get function for the name attribute	testAircraft	testAircraft	testAircraft	
1.03	Retreive aircraft origin	Aircraft.originName	Testing the get function for the name of the origin attribute of the aircraft	testAircraft	Berlin	Berlin	
1.04	Retreive aircraft destination	Aircraft.destinationName	Testing the get function for the name of the destination attribute of the aircraft	testAircraft	Dublin	Dublin	
1.05	Read if the plane has finished	Aircraft.isFinished	Testing the get funtion for the variable which reads TRUE if the aircraft has finished	testAircraft	FALSE	FALSE	
1.06	Read if the plane is being manually controlled	Aircraft.isManuallyControlled	Testing the get function for the variable that reads TRUE if the aircraft is in manual control mode	testAircraft	FALSE	FALSE	
1.07	Retreive the speed of the aircraft	Aircraft.speed	Testing the get function for the speed of the aircraft	testAircraft	20	20	
1.08	Retreive the altitude state of the aircraft	Aircraft.altitudeState	Calculating if the aircraft is acsending, descending or stationary	testAircraft 1	1	1	
1.09	Test if the aircraft is out of bounds	Aircraft.outOfBounds	Calculating if the aircraft is out of the visible airspace	testAircraft	TRUE	TRUE	
1.10	Setting the altitude state of the aircraft	Aircraft.setAltitudeState	Setting the current altitude state of the aircraft	testAircraft 1	1	1	
2.01	Store co-ordinates as a vector	Vector	Creating a new vector object by setting the x, y and z co-ordinates	[1.0, 1.1, 1.2]	vector.x = 1.0 vector.y = 1.1 vector.z = 1.2	vector.x = 1.0 vector.y = 1.1 vector.z = 1.2	
2.02	Retreive co-ordinates from a vector	Vector.X, Vector.Y, Vector.Z	Retreving the x, y and z co-ordinates from a vector object	[1.0, 1.1, 1.2]	vector.x = 1.0 vector.y = 1.1 vector.z = 1.2	vector.x = 1.0 vector.y = 1.1 vector.z = 1.2	
2.03	Calculate magnitude of a vector	Vector.magnitude	Calculating the magnitude of a vector from the x, y and z values	[1.0, 1.1, 1.2] [12.0, 16.0, 21.0]	3 29	3 29	
2.04	Test for vector equality	Vector.equals	Test equality of the stored vector and an input vector	[1.9, 2.2, 7.4] [1.9, 2.2, 7.4] [9, 4.2, 5.1] [9, 4.2, 5.0]	TRUE FALSE	TRUE FALSE	
2.05	Calculate magnitude squared of a vector	Vector.magnitudeSquared	Calculating the square of the magnitude of the vector	[1.0, 2.0, 2.0] [12.0, 16.0, 21.0]	9 841	9 841	
2.06	Normalise vector	Vector.normalise	Converting the vector to a normalised form	[1.0, 2.0, 2.0] [1.0, 4.0, 8.0]	[1/3, 2/3, 2/3] [1/9, 4/9, 8/9]	[1/3, 2/3, 2/3] [1/9, 4/9, 8/9]	
2.07	Scale a vector	Vector.scaleBy	Scaling the vector object by an input value	[1.0, 2.0, 3.0] 1 [1.0, 2.0, 3.0] -2	[1.0, 2.0, 3.0] [-2.0, -4.0, -6.0]	[1.0, 2.0, 3.0] [-2.0, -4.0, -6.0]	
2.08	Vector addition	Vector.add	Adding an input vector to the vector object	[2.0, 2.0, 4.0] [1.0, 3.0, 2.0] [6.0, 8.1, 16.0] [1.0, 2.0, 3.0]	[3.0, 4.0, 6.0] [7.0, 10.1, 19.0]	[3.0, 4.0, 6.0] [7.0, 10.1, 19.0]	
2.09	Vector subtraction	Vector.sub	Subtracting an input vector from the vector object	[2.0, 3.0, 4.0] [1.0, 1.0, 2.0] [14.0, 6, 100.0] [1.0, 6.0, 0.0]	[1.0, 2.0, 2.0] [13.0, 6.0. 0.0]	[1.0, 2.0, 2.0] [13.0, 6.0. 0.0]	
2.10	Angle between vectors	Vector.angleBetween	Calculating the angle between the object vector and an input bector	[1, 0, 0] [0, 1, 0]	π/2	π/2	
3.01	Initialise score	Score	Creating a new score object with default score values		score.timePlayed = 0 score.flights = 0 score.manualTime = 0 score.timeViolated = 0 score.gameOvers = 0	score.timePlayed = 0 score.flights = 0 score.manualTime = 0 score.timeViolated = 0 score.gameOvers = 0	Many of these attributes will not be in the release version
3.02	Retreive time played and successful flights	Score.timePlayed Score.flightsSuccessful Score.timeViolated Score.timeManual Score.gameOvers	Retrieving the score data from the object		score.timePlayed = 0 score.flights = 0 score.manualTime = 0 score.timeViolated = 0 score.gameOvers = 0	score.timePlayed = 0 score.flights = 0 score.manualTime = 0 score.timeViolated = 0 score.gameOvers = 0	Many of these attributes will not be in the release version
3.03	Adding time played	Score.addTime	Adding time played to the current length of session	6 4 1000 127	9 1127	9 1127	Test consit of adding time in sequence Test consit of adding time in sequence
3.04	Adding time in manual control	Score.addTimeManual	Adding time in manual mode to the current total manual mode length	8 4 14 88	12 102	12 102	Function was not used in final release Function was not used in final release
3.05	Adding seperation violation time	Score.addTimeViolated	Adding seperation volated time to the current value	10 16 0 1	26 1	26 1	Function was not used in final release Function was not used in final release
3.06	Adding a successful flight	Score.addFlight	Incrementing the number of successful flights	looped 10 times	10	10	Function was not used in final release
3.07	Adding a game over	Score.addGameOver	Incrementing the number of game overs	looped 13 times looped 4 times	13 4	13 4	Function was not used in final release Function was not used in final release
3.08	Calculate score	Score.calculate	Calculating the game score from the stored score values	gameOvers = 5 successfulFlights = 9 timeViolated = 9 timeManual = 10 timePlayed = 1000 gameOvers = 1 successfulFlights =100 timeViolated = 0 timeManual = 0 timePlayed = 1000	-3119 10000	-3119 10000	Function was not used in final release Function was not used in final release
4.01	Retreive waypoint position	Waypoint.position	Retreiving the position of the waypoint from the object	[10,10,0]	[10,10,0]	[10,10,0]	
4.02	Is mouse over waypoint	Waypoint.isMouseOver	Testing if the cursor is positioned over the waypoint	waypoint = [5,5] mouse = [10,10] waypoint = [5,5] mouse = [25,25]	TRUE FALSE	TRUE FALSE	
4.03	Test if entry/exit point	Waypoint.isEntryOrExit	Testing if the waypoint object is an entry/exit point	entryOrExit = FALSE entryOrExit = TRUE	FALSE TRUE	FALSE	
4.04	Calculate cost	Waypoint.getCost	Testing the calculation of the cost between the a waypoint and the current waypoint.	[2, 4, FALSE] [2, 2, TRUE] [6, 15, FALSE] [15, 15, TRUE]	2 9	2 9	
4.05	Calculate cost between	Waypoint.getCostBetween	Testing the calculation of the cost between the current waypoint and another waypoint.	[2, 4, FALSE] [2, 2, TRUE] [6, 15, FALSE] [15, 15, TRUE]	2 9	2 9	

Test Aircraft

The following is the test data used for many of the Aircraft class tests

Name	testAircraft
originName	Berlin
destinationName	Dublin
originWaypoint	100,100,True
destinationWaypoint	0,0 True
speed	10
waypointList	0, 0, True
	100, 100, True
	25, 75, False
	75, 25, False
	50, 50, False