Blackbox testing for function getBestRoute()
Prototype: int getBestRoute(struct Route routes[500], int s0, struct Shipment s);

Shipment and route will always be valid since previous functions will check validation for it

Return 1 = valid data

Return 0 = invalid data

2=blue

4=yellow

6=green

Function: getBestRoute	**Description**	**Test Data**	**Expected result**
Test1	Test to see if the size would be valid if the value is zero	int s0 = 0	Return 0
Test2	Test to see if the size would be valid if the value is -1	int s0 = -1	Return 0
Test3	Test to see if the size would be valid if the value is NULL	int s0 = NULL	Return 0
Test4	Test to see if the size would be valid if the value is ten	int s0 = 10	Return 1
Test5		struct shipment s{500,0.5, {7,9} //{H,9} struct Point p1[2] = { {8,10}, {7,10} }; struct Route r1= { p1, 2, 2}; struct Point p2[4] = { {4,10}, {5,10}, {6,10}, {7,10} }; struct Route r2 = {p2, 4, 4} struct Point p3[9]= { {10,5}, {10,6}, {10,7}, {9,7}, {9,8}, {8,8}, {8,9}, {8,10}, {7,10} }; struct Route r3 = {p3,9,6} struct Route routes[3] = {r1,r2,r3};	Return 1 Closest route is r1
Test6		struct shipment s{500,0.5, {10,3} //{K,3}	Return 1

		//blue
		struct Point p1[2] = { {9,4}, {10,4} };
		struct Route r1= { p1, 2, 2};
		
		//yellow
		struct Point p2[9] = { {3,5}, {3,4}, {4,4}, {5,4}, {6,4}, {7,4}, {8,4}, {9,4}, {10,4} };
		struct Route r2 = {p2, 9, 4}
		
		/green
		struct Point p3[1]= { {10,4} };
		struct Route r3 = {p3, 1,6}
		
		struct Route routes[3] = {r1,r2,r3};
			Closest route is r3

Test7		struct shipment s{500,0.5, {1,8} //{B,8}
		
		//blue
		struct Point p1[3] = { {2,6}, {2,7}, {2,8} };
		struct Route r1= { p1, 3, 2};
		
		//yellow
		struct Point p2[1] = { {2,8} };
		struct Route r2 = {p2, 1, 4}
		
		/green
		struct Point p3[11]= { {10,6}, {9,6}, {8,6}, {7,6}, {6,6}, {5,6}, {4,6}, {3,6}, {2,6}, {2,7}, {2,8} };
		struct Route r3 = {p3, 11,6}
		
		struct Route routes[3] = {r1,r2,r3};
			Return 1
			
			Closest route is r2