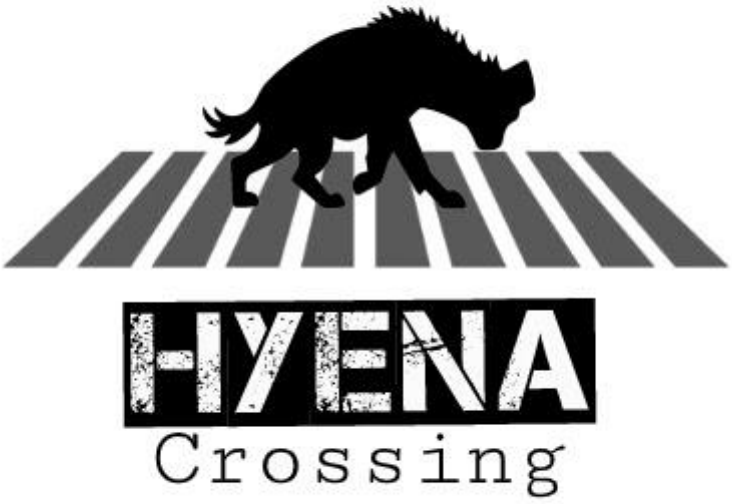
TEST REPORT



Yousef Abu Zahra

Borislav Pavlov

Bozhidar Bonev

Alina Baciu

Ghazi Abdul Fattah

Introduction

The Test Report document reflects data obtained from an evaluation experiment in an organized manner on the Traffic Lights Simulation, describing the environmental or operating conditions, and showing the comparison of test results with test objectives.

Testing Objectives

The purpose of this testing is to verify the functionality of all components.

We are going to test the following features:

Simulation Runtime Controls

- Start simulation
- Stop simulation
- Pause simulation
- Generate report

Map Layout

- Add/Remove road
- Reset Map Layout

File options

- Save a file
- Open an existing file

Dynamic elements

- Specify number of cars generated
- Specify environment
- Reset default values
- Change simulation speed
- Dijkstra Algorithm

Testing Approach

For now, we are going to use Unit Testing which is a method that instantiates a small portion of our application and verifies its behavior independently from other parts. A unit test can be manual (performed by people) and automated (which performs the process without human intervention).

A typical automated unit test contains 3 phases: First, it initializes a small piece of an application it wants to test, then it applies some stimulus to the system under test (usually by calling a method on it), and finally, it observes the resulting behaviour. If the observed behavior is consistent with the expectations, the unit test passes, otherwise, it fails, indicating that there is a problem somewhere in

the system under test. These three-unit test phases are also known as Arrange, Act and Assert, or simply AAA.

Throughout this document both types of the unit testing will be used in order to show the accuracy of each functionality.

Simulation Runtime Control

Start simulation

Test case ID: 1

Test case: check whether it is possible to start the timers of the application

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
1.a	Start simulation– Automated Testing	Program is launched.	1.Grid object is created 2. StartTimers() method is called	Grid g = new Grid(3,6);	True. The timer of the car and road starts.	✓	-
1. b	Start simulation– Manual Testing	Program is launched. Vehicles per lane need to be specified. At least one type of road is added. Planned Simulation time is added.	1.User presses on Play button	Data from the running application	Cars appear on screen and simulate real traffic time.	✓	-

Stop simulation

Test case ID: 2

Test case: check whether it is possible to stop the timers of the application

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
2.a	Stop simulation – Automated Testing	Program is launched.	1.Grid object is created 2.StartTimers() method is called 3.StopTimers() method is called	Grid g = new Grid(3, 6);	True. The timer of the car and road stops.	✓	-
2.b	Stop simulation – Manual Testing	Program is launched. Vehicles per lane need to be specified. At least one type of road is added. Planned Simulation time is added. Start button needs to be pressed.	1.User presses on Stop button	Data from the running application	The simulation is stopped.	✓	-

Pause simulation

Test case ID: 3

Test case: check whether it is possible to pause the timers of the application

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
3.a	Pause simulation – Automated Testing	Program is launched.	1.Grid object is created 2.StartTimers() method is called 3. PauseTimers() method is called	Grid g = new Grid(3, 6);	True. All timers pause.	✓	-

3.b	Pause simulation – Manual Testing	Program is launched. Vehicles per lane need to be specified. At least one type of road is added. Planned Simulation time is added. Start button needs to be pressed.	1.User presses on Pause button	Data from the running application	The simulation is paused.	✓	-

Map Layout

Add road

Test case ID: 4

Test case: checks whether it is possible to add roads in simulation

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
4.a	Add road – Automated Testing	Program is launched.	1.Grid is initialized 2. AddRoad() method is called 2 times	f.ReturnGrid().AddRoad(new Road(new Point(4, 5), pb_b), 1); f.ReturnGrid().AddRoad(new Road(new Point(4, 6), pb_b), 2);	The number of the current placed roads in the grid is 2.	✓	-
4. b	Add road – Manual Testing	Program is launched.	1.User chooses grid size 2. User drags and drop chosen road	gridPanel_DragDrop(object sender, DragEventArgs e); PointToClient(Cursor.Position) ;	The chosen road is added and dropped on the right spot on the grid	✓	-

Remove road

Test case ID: 5

Test case: - verifies whether removing a non-existing road works

- checks whether it is possible to remove roads in simulation

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
5.a	Remove Road – Automated Testing	Program is launched.	1.Grid is initialized 2.RemoveCrossing(2) method is called	f.ReturnGrid().RemoveCrossing (2)	False. It is not possible to remove a nonexistent road.		-
5.b	Remove Road – Manual Testing	Program is launched.	1.User right clicks on chosen road 2. User chooses „remove Crossing“option	removeCrossingToolStripMenuItem_Click(object sender, EventArgs e)	The road selected by the user will be removed from the grid.		-

Reset Map Layout

Test case ID: 6

Test case: check whether it is possible to clear the grid of the application

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
6.a	Clear screen – Automated Testing	Program is launched.	1. Grid is initiliazed 2. AddRoad() method is called 3. ReturnPictureBoxes().Add() method is called 4. ClearGrid() is called	AddRoad(new Road(new Point(5, 3), pb_b), 1)	The number of pictures and roads returned is 0.		-
6.b	Clear screen– Manual Testing	Program is launched. The grid is not empty.	1.User presses on Clear All button	Data from running simulation	The grid is empty.		-

File Options

Save data

Test case ID: 7.

Test case: check if the grid of the simulation is serialized and saved using random path

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
7a.	Save file – Automated testing	Program is launched. Simulation is paused.	1.Grid is initiliazed 2.Save() method is called	f.Save(@"D:\TestExample1")	File with data from paused simulation.	✓	-
7b.	Save file- Manual testing	Program is launched. Simulation is paused.	1.User presses on Save button 2. User gives location and presses save	Save(saveas.FileName)	File with data from paused simulation.	✓	-

Open an existing file

Test case ID: 8.

Test case: open an existing file and check if the grid shown has roads and pictures

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
8a.	Open File – Automated testing	Program is launched.	1.Load() method is called	f.loadFile(@"D:\TestExample1");	The returned grid has roads and pictures	✓	-

8b.	Open file-Manual testing	Program is launched.	1.User clicks on open file button	loadFile(openFile.FileName);	Existing runnable project with roads and pictures		Problem that needs to be fixed: The planned time is not serialized and it has to be given by the user every time

Dynamic Elements

Specify number of cars generated

Test case ID: 9

Test case: A. Verify that roads have the proper amount of cars generated on them

B. Verify that roads are created with no cars

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
9.a	Verify amount of cars generated on lane	Program is launched.	1.Create road object 2.Add x amount of cars to a lane of the road 3.Check if the roads have the amount of cars specified	Road road = new RoadType1(new Point(0, 0), new PictureBox()); road.AddCars(road.Lanes[0],5);	Roads should have the amount specified, and the sum of all cars on that road lanes should be that amount too.		-
9.b	Verify roads are created with no cars	Program is launched.	1.Create road object 2.Check if nr. of cars equals to 0	Road road = new RoadType1(new Point(0, 0), new PictureBox());	Road is created with no cars.		-

Reset default values

Test case ID: 10

Test case: Reset the default values of textboxes in the form

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
10 a.	Reset default values - Automated testing	Program is launched.	1.Initial values are given 2.ResetValues() method is called	form.tbVehicelsAmount.Text = "1"; form.pdstrlower.Text = "2"; form.pdstrupper.Text = "3";	Clear the values of the textboxes.	✓	-

Change simulation speed

Test case ID: 11

Test case: A. Verify whether it is possible to change the simulation speed

B. Verify whether giving invalid value to trackbar throws exception

C. Verify whether giving invalid value to grid throws exception

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments

11 a.	Change simulation speed	Program is launched.	1.Create a road object
			2. Change the value of the trackbar
			3.Check whether the simulation speed is equal to the new value of the trackbar
				AddRoad(new Road(new Point(5, 3), pb_b), 1);
				
				f.TrackBarSpeed.Value = 5;
				
				f.TrackBarSpeed.Value = 10;
					The simulation speed is equal to the value of the trackbar.	✓	-

11 b.	Change trackbar simulation speed exception	Program is launched.	1.Give invalid value to the trackbar	f.TrackBarSpeed.Value =0;	Exception	✓	-
11 c.	Change grid simulation speed exception	Program is launched.	1.Grid is initialized 2.Give invalid value to the simulation speed of the grid	f.ReturnGrid().SimSpeed = 0;	Exception	✓	-

Dijkstra Algorithm

Test case ID: 12

Test case: A. Test if the algorithm actually gives out the shortest path from start to target

B. Check if we get all possible points and lanes on the grid

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments
12. a	Check shortest path	Program is launched.	1.We create the points 2.We add the points to nodes 3.We add the nodes to the tracker 4.We connect the nodes 5. We get the shortestpath and the actual shortestpath 6. We check if they are equal	// ----- Creating the points Point start = new Point(0, 0); Point p2 = new Point(200, 300); Point p3 = new Point(150, 200); Point p4 = new Point(500, 100); Point p5 = new Point(50, 500); Point target = new Point(200, 200); // ---------- Adding the points to nodes ---------- Node startNode = new Node(start); Node endNode = new Node(target); Node n2 = new Node(p2);	actualShortestPath.Count=shortestPath.Count actualShortestPath[0]=shortestPath[0] actualShortestPath[1]=shortestPath[1] actualShortestPath[2]= shortestPath[2]	✓	-

				Node n3 = new Node(p3); Node n4 = new Node(p4); Node n5 = new Node(p5);			
12. b	Check points and lanes on grid	Program is launched.	1.Grid is initialized 2. We add a test road type 1 to the grid 3. We get all points on the grid 4.We check whether the number of the lanes and points is equal to the nr of lanes and points of road type 1	Grid myGrid = new Grid(3, 3); Road testRoad = new RoadType1(new Point(0, 0), new PictureBox()); myGrid.AddRoad(testRoad, 0);	// Note: RoadType 1 have 12 lanes, with 1 starting point each Assert.AreEqual(12, allPoints.Count); Assert.AreEqual(12, pointSum);		-

Specify Environment

Test case ID: 13

Test case: A. Checks whether the background color in form1.cs is changing when button night mode is triggered

B. Checks whether the background color in form1.cs is changing when button day mode is triggered

ID	Test name	Preconditions	Test steps	Test data	Expected result	Pass	Comments

13. a	CheckNightMode	Program is launched	1. Create form object
			2. Calling event button click method
				var frm = new Form1();
				frm.envtwobtn_Click(this, null);
				Assert.AreEqual(Color.DarkSlateBlue, frm.BackColor);
					The actual color is match to the background color of night		-

13. b	CheckDay Mode	Program is launched	1. Create form object
			2. Calling event button click method
				var frm = new Form1();
				frm.envonebtn_Click(this, null);
				Assert.AreEqual(Color.White, frm.BackColor);
					The actual color is match to the background color of day		-