Test Plan

1. **Introduction**

This project is related to the delivery system that will be used by local delivery companies. We basically receive information from our customers, such as the weight and size of the shipment, and the destination of the delivery. Based on this information, the goal of this program is to find the truck with the best conditions and its route. Every truck has a fixed weight and volume. Each truck has its own truck with three different routes. Each truck can deliver through its designated route, or through space where no other truck's routes or buildings exist. Based on these conditions, the shortest possible distance for a truck is calculated. Compare the calculated distances to find the best route for the truck. If the distances are the same, we will find a truck that currently carries fewer shipments. Our test will produce our expected results by examining the conditions to be tested in this process, such as, 'Does this shortest distance pass through the space where a building exists?'

1. **Scope**
2. What will be tested.

i. Whether the calculated route is crossing only the space that the truck can pass through

ii. Whether the route reach a given destination

iii. If the shipment is loaded, whether it does not exceed the maximum loading weight or volume

b. What will not be tested.

i. No test is conducted assuming a destination outside the 25 by 25 square grid

1. **Test Strategy**

**System Test**

The system should print the required statement as the sample output

The system should wait for the user to enter their input

The buffer should receive 3 inputs from the users including shipment weight, box size, and destination

The system should announce user if the user inputs an invalid input

The system should announce if any of the 3 inputs out of the limit range

…. no diversion

If all inputs are ok, the system has to display the path for the users

The output should show the closest line with the shortest path to delivery.

The system should exit when the user inputs “0 0 x”

**Performance Test**

The test should show the result immediately when the user hits enter

unit test

analysis of all the function

=> testing if all the functions meet all the requirements

test the function in the code to see if it works

using all techniques like print statements, debugging,

You could describe the test design process and give an overview of how it will be conducted. You could provide a broad overview of

1. how to understand requirements,

-show an invalid message if the user inputs invalid data

-calculate the fastest path to delivery

-if there is no space for the fastest line, it will change to the next line

-

1. build a traceability matrix,

-create 7 business requirements

1. prepare test cases,

-test negative:

-test positive:

-test out of range:

-test unit

-test integration

1. and have them reviewed by another member of the quality assurance team.
2. **Environment Requirements**

**Software requirement:**

Web Browsers: Chrome (latest version)

Development Environment: Visual Studio Code 2020

Programming Language: C++, C

Version Control System: Git

Planning system: Jira planner

**Hardware requirement:**

Windows 10 or higher, 1.6 GHz or faster processor, 1 GB RAM

**network requirement:**

Wi-fi, 4G, 5G, LTE

**Testing Tools:**

Visual Studio Code 2020 Debugging

**Test Harness:**

log file

assertion

lint

print statement

1. **Execution Strategy**

The severity of defects and break them down into severity levels of:

1. **critical**: before displaying output, the program stops running because of error
2. **high**: Unless one of the testing(or more) results as we expected, it would be considered this level.
3. **medium**: All of the testing results as we expected, but output seems not correct.
4. **Low**: Although the output results are logically consistent, there are spelling errors that interfere with the user's understanding.
5. **Cosmetic**: **X**
6. **Test Reporting**

The test is conducted when modifications are made to the software, such as when adding new code or removing existing code. At the beginning of the project, after all the necessary logic is expressed, the test is conducted for the first time.

After that, tests are conducted after all modification steps to determine whether the software works properly.

The results of the test are recorded where the team members can see (git), and a message is sent to inform the team members that they uploaded the record to the Teams chat to which they belong.

1. **Test Schedule**

As mentioned in "Testing Reporting", the first test is done when all the necessary logic is first written. And after modifying the code, the test is always conducted additionally to evaluate its impact.

The first test will take a lot of time to evaluate all the logic.

Testing after partial modification may be allowed to proceed by selecting a test that is expected to affect the modified part. In this case, it is expected to take less time compared to the first test.

If the results of the test show something unexpected, the test will repeat while finding the reason and correcting the code, and it is difficult to predict the time it takes for this test.

1. **Control Procedures**

If a bug or a point to be corrected is found, it should be shared with any one of the git, jira, or teams chat. If it is resolved during individual working, share it to members.

Basically, discuss this issue during a set team meeting. If the problem is not resolved within the allowable time, schedule the next meeting and wrap up the meeting.

Even if it is not because of a bug or an error, all the fixes must be shared.

1. **Functions To Be Tested**
2. **Resources and Responsibilities X**  
   8.1. Resources  
   8.2. Responsibilities
3. **Deliverables X**
4. **Suspension / Exit Criteria X**
5. **Resumption Criteria X**
6. **Dependencies X**  
   12.1 Personnel Dependencies  
   12.2 Software Dependencies  
   12.3 Hardware Dependencies  
   12.3 Test Data & Database
7. **Risks X**  
   13.1. Schedule  
   13.2. Technical  
   13.3. Management  
   13.4. Personnel  
   13.5 Requirements
8. **Tools X (It’s already written above)**
9. **Documentation X**

**Approvals X**