**Group-11**

**SFT221**

**Acceptance Test Document**

Overview

This acceptance test document aims to ensure the correctness and functionality of the C code provided in cargo.c and crago.h . The tests cover different scenarios to confirm that the functions perform as expected.

Test Environment

* **Operating System:** Windows 10
* **Compiler:** GCC
* **Libraries:** **stdio.h**

Test Cases

1. Printing Header (printHdr())

Objective: Confirm that the header is printed correctly.

Input: No input needed.

Expected Output: Seneca College Deliveries

**2. Printing Footer (printFtr())**

* **Objective**: Verify if the footer is printed correctly.
* **Input**: No input required.
* **Expected Output**: Thanks for shipping with Seneca!

**3. Setting Truck Information (setTruck(struct TruckInfo\* truck))**

* **Objective**: Ensure that the truck weight and box size are set correctly.
* **Input**:
  + Enter truck weight: 1200kg
  + Enter truck box size: 1.5
* **Expected Output**: No output. The function should set the truck weight to 1200 kg and the box size to 0.5,1,5 cubic meters.

4. Validating Destination (isDestValid(const struct Map\* routeMap, int destRow, char destCol))

Objective: Confirm if a destination is valid based on the route map.

Input:

Valid destination coordinates: (3, 'C')

Expected Output: true

5. Checking Destination Input (checkDestInput(const char\* inputString, int\* rowNum, char\* columnChar))

Objective: Verify if the destination input string is parsed correctly.

Input:

Destination input string: "3C"

Expected Output: true

6. Validating Weight (validateWeight(double weight))

Objective: Confirm if a given weight falls within the valid range.

Input:

Valid weight: 500

Expected Output: true

7. Validating Box Size (validateBoxSize(double boxSize))

Objective: Confirm if a given box size is valid.

Input:

Valid box size: 0.5

Expected Output: true

8. Getting Symbol from Route Map (getSym(const struct Map\* routeMap, int row, char column))

Objective: Retrieve the symbol at the specified coordinates on the route map.

Input:

Coordinates: (5, 'E')

Expected Output: [Symbol]

Converting to Coordinate (convertToCoordinate(int row, char column))

Objective: Convert row and column to coordinate struct.

Input:

Row: 5, Column: 'E'

Expected Output: {5, 'E'}

10. Finding Closest Coordinate (findClosest(const struct Coordinate destination, const struct TruckInfo\* trucks, const struct Map\* baseMap))

Objective: Find the closest coordinate to the destination.

Input:

Destination: (10, 'K')

Expected Output: {10, 'J'}

11. Printing Diversion Route (printDiversion(const struct Route\* diversionRoute, const struct Route\* originalRoute, const struct Coordinate\* destination))

Objective: Print the diversion route.

Input:

Diversion Route: Route with diversion points

Original Route: Original route points

Destination: Final destination

Expected Output: (7, H) (8, H) (9, H) (10, H) (10, I) (10, J)

Conclusion

All test cases in the acceptance testing document have passed successfully, indicating that the functions implemented in the cargo.h library behave as expected and meet the requirements outlined in the project description.

Overall, the acceptance testing process confirms the robustness and reliability of the software, ensuring its readiness for deployment in the local delivery company's operations. Further refinement and optimization may be considered in future iterations, but the current implementation meets the specified criteria effectively.