|  |  |
| --- | --- |
| A picture of a winding road and trees  Function Specifications | Abstract  The purpose of this document is to provide all the information of the functions to start writing Blackbox test casing as well as giving guidelines to the developers.  Jarod Jian Kang Hery Chen  SFT Milestone 3 |

**NAME:** destinationFinished **Description:** Determines if the destination is reached by comparing the distance between two points.

**Parameters:**

* const struct Point p1 – The first point.
* const struct Point p2 – The second point.

**Returns:** 1 if the distance between the points is 1, otherwise 0.

**NAME:** isWeight **Description:** Checks if the weight is within the valid range.

**Parameters:**

* int weight – The weight to check.

**Returns:** 1 if the weight is within the valid range, otherwise 0.

**NAME:** isSize **Description:** Checks if the size is a valid value.

**Parameters:**

* double size – The size to check.

**Returns:** 1 if the size is valid, otherwise 0.

**NAME:** isValidShipment **Description:** Validates the weight and size of the shipment.

**Parameters:**

* int weight – The weight of the shipment.
* double size – The size of the shipment.

**Returns:** 1 if both weight and size are valid, otherwise 0.

**NAME:** conversion **Description:** Converts a string representation of a destination to a Point structure.

**Parameters:**

* const char arr[] – The string representing the destination.

**Returns:** A struct Point representing the destination.

**NAME:** convertDestination **Description:** Converts a Point structure to a string representation of the destination name.

**Parameters:**

* const struct Point pt – The point to convert.

**Returns:** A string representing the destination name.

**NAME:** isValidDestinationName **Description:** Checks if the destination name is valid.

**Parameters:**

* const char destinationName[] – The destination name to check.

**Returns:** 1 if the destination name is valid, otherwise 0.

**NAME:** findAlpha **Description:** Converts a character to the corresponding index.

**Parameters:**

* char c – The character to convert.

**Returns:** The index corresponding to the character, or -1 if the character is invalid.

**NAME:** getCapacity **Description:** Checks which trucks can handle the shipment based on their current capacity.

**Parameters:**

* struct Truck trucks[] – Array of trucks.
* int numTruck – The number of trucks.
* struct Shipment pkg – The shipment to be loaded.

**Returns:** An array of integers indicating which trucks can handle the shipment.

**NAME:** getDistanceToDestination **Description:** Calculates the distance from the route to the destination.

**Parameters:**

* struct Route\* route – The route of the truck.
* struct Point dest – The destination point.

**Returns:** The distance to the destination.

**NAME:** getRoute **Description:** Retrieves the route based on the truck symbol.

**Parameters:**

* char symbol – The symbol of the truck route.

**Returns:** A pointer to the struct Route of the truck.

**NAME:** findBetterCapacityTruck **Description:** Compares the capacities of two trucks to determine which has better capacity.

**Parameters:**

* const struct Truck\* truck1 – The first truck.
* const struct Truck\* truck2 – The second truck.

**Returns:** 1 if the second truck has better capacity, otherwise 0.

**NAME:** handleDiversion **Description:** Handles the diversion of a truck to avoid obstacles, calculating a new route.

**Parameters:**

* const struct Route\* selectedRoute – The current route of the truck.
* struct Point dest – The destination point.

**Returns:** A struct Route representing the new route.

**NAME:** findBestTruckShipment **Description:** Finds the best truck for the shipment based on weight, size, and destination.

**Parameters:**

* struct Truck\* trucks – Array of trucks.
* const int weight – The weight of the shipment.
* const double size – The size of the shipment.
* const char destination[] – The destination name.

**Returns:** A pointer to the best truck for the shipment, or NULL if no suitable truck is found.

**NAME:** validInput **Description:** Validates the user input for the shipment's weight, size, and destination name.

**Parameters:**

* const int weight – The weight of the shipment.
* const double size – The size of the shipment.
* const char destination[] – The destination name.
* struct Truck\* trucks – Array of trucks.

**Returns:** None. Prints messages if the input is invalid.