**Specification of functions that are added in ‘Shipment.h’**

**NAME:** init

**Description:** Initializes global variables that will be used throughout the project. So, these functions provide control for assigning and storing values to variables.

**Parameters:**

* No Parameters

**Returns:** No return values

**NAME:** validate

**Description:** Validates user input for shipment that includes weight (1-1000 kgs), size (0.25, 0.50, 1.00), and destination (0A). If all inputs are valid, it will return 1 and if exit code (0 0 x) is entered than returns 0.

**Parameters:**

* size - size of shipment: S\_SIZE, M\_SIZE, or L\_SIZE
* weight - weight of shipment in kgs (1 - 1000)
* destination - shipment's delivery destination from {0,0} to {25,25}

**Returns:**

1 if all values are valid,

-1 if invalid size,

-2 if invalid weight,

-3 if invalid destination,

0 if exit code is entered.

**NAME:** read

**Description:** Reads user input for new shipments and sends the values into validate function for validation.

**Parameters:**

* No Parameters

**Returns:** No return values

**NAME:** getAllTruckPaths

**Description:** populates array “routes” with routes that reaches to the destination and store it in struct.

**Parameters:**

* s - Shipment struct, containing the Shipment destination
* truck - Truck struct, containing the route the truck travels on the map
* map - Map struct, containing all inbound points in the map
* routes - holds all valid routes (i.e. struct Route)

**Returns:** No return values, populates the "routes" array

**NAME:** getBestRoute

**Description:** loops through all valid routes in "routes" array that were populates by getAllTruckPaths functions and returns index of route with the shortest distance.

**Parameters:**

* routes - holds all valid routes (i.e. struct Route)
* s - Shipment struct, containing the Shipment destination

**Returns:** int, an index of route in "routes" array with the shortest route

**NAME:** containsDestination

**Description:** checks if the given route is valid if it reaches the destination and if destination is within route.

**Parameters:**

* route - the shortest route calculated at ith point of a trucks route
* s - Shipment struct, containing the Shipment destination

**Returns:** int, 1 if the destination is in the route, 0 if not

**NAME:** printRoute

**Description:** print route to console just like printMap.

**Parameters:**

* route - route struct containing the points in given route
* s - Shipment struct, containing the Shipment destination

**Returns:** No return values

**NAME:** isBuilding

**Description:** checks for building location at all points on route, if there is any building, return building’s index.

**Parameters:**

* r - route struct containing the points in given route
* map - Map struct, containing all inbound points in the map

**Returns:** int, 1 if there is a building, 0 if not.

**NAME:** AssignTruck

**Description:** Determine best truck for shipment to load and assign it the best route.

**Parameters:**

* s - the Shipment to deliver

**Returns:** the truck assigned to shipment s by color code, returns 2 if shipment is assigned to the blue truck, returns 4 if shipment is assigned to the green truck, returns 8 if shipment is assigned to the yellow truck, returns -1 if shipment cannot be delivered today

**NAME:** calculateUtilizationScore

**Description:** Determines the utilization score based on defined weight and volume limit

**Parameters:**

* weight - the truck's current weight or shipment's weight
* vol - the truck's current volume or shipment's volume

**Returns:** float, 0 if inputs are invalid, otherwise floating value