# Function Description

**Function Name:** hasEnoughSpace

**Parameter List:**

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| tr | const struct Truck\* | Pointer to the truck to check capacity |
| wght | const int | Weight of the shipment to be loaded |
| vol | const int | Volume of the shipment to be loaded |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Returns:** ‘int’ - Returns 1 if the truck can hold the weight and volume, 0 if not.

**Description:** This function checks if the truck, represented by the pointer tr, has enough remaining capacity to hold a shipment with the specified weight (wght) and volume (vol). It compares the truck's current load and capacity against the new shipment's requirements. If the truck can accommodate the additional weight and volume, it returns 1. Otherwise, it returns 0.

**Function Name:** loadShipment

**Parameter List:**

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| tr | struct Truck\* | Pointer to the truck to load the shipment into |
| wght | const int | Weight of the shipment to be loaded |
| vol | const int | Volume of the shipment to be loaded |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Returns:** ‘void’

**Description:** This function modifies the truck's current load by adding the specified weight (wght) and volume (vol) of the shipment. It updates the truck's attributes to reflect the new total weight and volume after loading the shipment. This function assumes that the truck has enough capacity to hold the shipment and does not perform any validation.

**Function Name:** checkMoreCapacity

**Parameter List:**

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| tr1 | const struct Truck\* | Pointer to the first truck for comparison |
| tr2 | const struct Truck\* | Pointer to the second truck for comparison |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Returns:** Returns 1 if the first truck (tr1) has more capacity, 2 if the second truck (tr2) has more capacity.

**Description:** This function compares the remaining capacity of two trucks (tr1 and tr2). It calculates which truck has more available space for additional weight and volume. If the first truck (tr1) has more remaining capacity, it returns 1. If the second truck (tr2) has more remaining capacity, it returns 2. This function helps determine which truck is better suited for carrying additional shipments.

**Function Name:** validateBox

**Parameter List:**

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| size | const int | Size of the box to validate |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

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

**Description:** This function checks if the given box size (size) is within acceptable limits for loading into a truck. It validates the box dimensions based on predefined criteria (e.g., maximum allowed size). If the box size is valid, it returns 1. Otherwise, it returns 0. This function ensures that only appropriately sized boxes are loaded into trucks.

**Function Name:** validateDest

**Parameter List:**

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| dest | const struct Point\* | Pointer to the destination point to validate |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

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

**Description:** This function checks if the given destination (dest) is valid for delivery. It validates the destination based on certain criteria (e.g., within delivery range, accessible location). If the destination is valid, it returns 1. Otherwise, it returns 0. This function ensures that shipments are only sent to appropriate and reachable destinations.

**Function Name:** validateWeight

**Parameter List:**

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| weight | int | The weight of the shipment to validate |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Returns:** Returns 1 if the weight is valid (greater than 0), 0 otherwise.

**Description:** This function checks if the provided weight for a shipment is valid. It ensures that the weight is a positive value. If the weight is greater than 0, the function returns 1, indicating the weight is valid. If the weight is less than or equal to 0, it returns 0, indicating an invalid weight. This validation is essential to ensure that only shipments with appropriate weights are processed.

**Function Name:** validateShipment

**Parameter List:**

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| ship | const struct Shipment\* | Pointer to the Shipment structure to be validated. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

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

**Description:** This function validates the given shipment by checking its size, weight, and destination. It integrates three separate validation checks:

validateBox: Checks if the size of the shipment is valid (matches predefined box sizes).

validateWeight: Checks if the weight of the shipment is valid (greater than 0).

validateDest: Checks if the destination of the shipment is within the bounds of the map.

If all three validations pass, the function returns 1, indicating the shipment is valid. If any validation fails, it prints an appropriate error message ("Invalid size!", "Invalid weight! (must be 1-2500 Kg)", or "Invalid destination!") and returns 0. This comprehensive validation ensures that shipments processed are within acceptable parameters for size, weight, and destination.