Package greedy_algorithm

Class Summary

CargoPair

This class models a (hardware, units) - pair.

Coordination

This class Coordinates the loading process of the respective trucks under the specifications defined in the Main Method.

Item

This class models the hardware parts that are needed.

Truck

This class represents the truck that will transport the goods to Bonn.

Warehouse

This class represents a warehouse where the given hardware goods are stored.

greedy_algorithm

Class CargoPair

```
java.lang.Object
   |
   +--greedy_algorithm.CargoPair
```

All Implemented Interfaces:

java.lang.Comparable

< Fields > < Constructors > < Methods >

public class **CargoPair**
extends java.lang.Object
implements java.lang.Comparable

This class models a (hardware, units) - pair. This is to make it easier to store the ordered quantity of an item or the stowed quantity of an item in an array.

Author:

Lea

Fields

amount

`public int amount`

---

item

```
public final Item item
```

Constructors

CargoPair

```
public  CargoPair(Item item,
                  int amount)
```

Methods

compareTo

```
public int compareTo(CargoPair o)
```

---

loading_weight

```
public int loading_weight()
```

Returns:

The total weight resulting from the quantity and the weight of the item.

---

greedy_algorithm

Class Coordination

```
java.lang.Object
    |
    +--greedy_algorithm.Coordination
```

< Constructors > < Methods >

---

public class **Coordination**
extends java.lang.Object

This class Coordinates the loading process of the respective trucks under the specifications defined in the Main Method.

Author:

Lea

Constructors

Coordination

```
public  Coordination()
```

Methods

change_last_item

```
public static void change_last_item(Warehouse warehouse,
                                    Truck truck)
```

Swaps the last item if the remaining space allows an item of higher value to be stowed.

Parameters:

warehouse -
truck -

---

fill_truck

```
public static void fill_truck(Warehouse warehouse,
                              Truck truck)
```

Fill a given truck with the available items in the warehouse regarding the priority and weight of the items.

Parameters:

warehouse -
truck -

---

get_fitting_units

```
public static CargoPair get_fitting_units(Item item,
                                          Warehouse warehouse,
                                          Truck truck)
```

It determines how many units of the given item would fit in the truck without checking the actual quantity available.

Parameters:

item -

Returns:

A CargoPair of item and units

---

given_information

```
public static java.util.ArrayList given_information()
```

creates the Item Objects with the required quantity with the information from the PDF

Returns:

order list for the Warehouse

---

main

```
public static void main(java.lang.String[] args)
```

---

print_cargo

```
public static void print_cargo(java.util.ArrayList cargo)
```

Printing the given cargo

Parameters:

cargo -

---

print_transport_value

```
public static void print_transport_value(Truck fst_truck,
                                         Truck snd_truck)
```

Prints the loaded value of the two trucks

Parameters:

fst_truck -
snd_truck -

---

greedy_algorithm

Class Item

```
java.lang.Object
    |
    +--greedy_algorithm.Item
```

< Fields > < Constructors > < Methods >

---

public class **Item**
extends java.lang.Object

This class models the hardware parts that are needed. The commodity is composed of the attributes name, weight and (utility-)value.

Author:

Lea

Fields

name

```
private java.lang.String name
```

value

```
private int value
```

weight

```
private int weight
```

Constructors

Item

```
public  Item(java.lang.String name,
             int weight,
             int value)
```

Methods

equals

```
public boolean equals(java.lang.Object obj)
```

Overrides:

equals in class java.lang.Object

getName

```
public java.lang.String getName()
```

---

getValue

```
public int getValue()
```

---

getWeight

```
public int getWeight()
```

---

setName

```
public void setName(java.lang.String name)
```

---

setValue

```
public void setValue(int value)
```

---

setWeight

```
public void setWeight(int weight)
```

---

greedy_algorithm

Class Truck

```
java.lang.Object
    |
    +--greedy_algorithm.Truck
```

< Fields > < Constructors > < Methods >

---

public class **Truck**

extends java.lang.Object

This class represents the truck that will transport the goods to Bonn. The weight of the driver and the capacity are taken into account here.

Author:

Lea

Fields

cargo

```
private java.util.ArrayList cargo
```

---

remaining_space

```
private double remaining_space
```

Constructors

Truck

```
public  Truck(double drivers_weight,
              double capacity)
```

Methods

can_Item_be_loaded

```
public boolean can_Item_be_loaded(Item item)
```

Checks if the weight of the given item exceeds the remaining space.

Parameters:

item -

Returns:

true if the item can be loaded into the truck

---

getCargo

```
public java.util.ArrayList getCargo()
```

---

getRemaining_space

`public double **getRemaining_space**()`

---

get_loaded_value

`public int get_loaded_value()`

After the truck is loaded, this method calculates the sum of all values of all items to get the whole transported value.

Returns:

The loaded value in the truck

---

load_Pair

`public void load_Pair(CargoPair pair)`

Adding the given pair to #cargo and updating the remaining space in the truck.

Parameters:

pair - hardware item and its amount in the cargo

---

setCargo

`public void setCargo(java.util.ArrayList cargo)`

---

setRemaining_space

`public void setRemaining_space(double remaining_space)`

---

greedy_algorithm

Class Warehouse

```
java.lang.Object
    |
    +--greedy_algorithm.Warehouse
```

< Fields > < Constructors > < Methods >

---

public class **Warehouse**
extends java.lang.Object

This class represents a warehouse where the given hardware goods are stored. The quantity ordered can be read from the order_list.

Author:

Lea

Fields

order_list

```
private java.util.ArrayList order_list
```

Constructors

Warehouse

```
public  Warehouse(java.util.ArrayList order_list)
```

Constructor will initialize the order_list sorted by priority.

Parameters:

order_list -

Methods

add_orders

```
public void add_orders(java.util.ArrayList new_orders)
```

The given orders won't be inserted by priority. The method will only concatenate the ArrayList with the already existing order_list.

Parameters:

new_orders -

---

clean_up_orders

```
public void clean_up_orders()
```

Cleaning up the {@link #order_list} with the help of the removed_items list. The names are uniquely! The method should be used after filling a truck.

---

extractItem

`public void extractItem(CargoPair cargo_pair)`

Updating the order quantity in the order_list to load a certain amount of an item type. The method compares the quantity to be loaded with the available quantity and updates the load if necessary. If the entire amount of an item type has been loaded, its index will be added to the #removed_items list.

Parameters:

pair - item to load and its updated amount

---

find_Index_of_Item

`public java.lang.Integer find_Index_of_Item(Item item)`

Parameters:

item -

Returns:

The Index of the CargoPair in the warehouse. If the CargoPair is no longer available, the method returns null.

---

getOrder_list

`public java.util.ArrayList getOrder_list()`

---

setOrder_list

`public void setOrder_list(java.util.ArrayList order_list)`

---

sort_by_priority

`public void sort_by_priority()`