We are preparing an application for managing container loading. Containers can later be transported by various types of vehicles - ships, trains, trucks, etc.

The system we are designing will handle the loading of containers onto a container ship - a vessel equipped with special guides allowing for the transport of containers. Containers can be of different types depending on the cargo. Bananas should be transported in refrigerated containers; milk should be transported in liquid containers; helium should be transported in gas containers. All these containers have some common features:

All containers have:

- The mass of the cargo (in kilograms)
- Height (in centimeters)
- Tare weight (the weight of the container itself, in kilograms)
- Weight of the cargo itself
- Depth (in centimeters)
- Serial number
  - The format of the number is KON-C-1
  - The first part of the number is always "KON"
  - The second part represents the type of container
  - The third part is a number. Numbers should be unique. There should be no possibility of two containers having the same number. Numbers should be generated by the system.
- The maximum payload of a given container in kilograms

All containers should allow for:

- Emptying the cargo
- Loading the container with a given mass of cargo
  - If the mass of the cargo is greater than the capacity of a given container, we should throw an OverfillException error

Liquid containers (L)

Liquid containers allow for the transportation of hazardous cargo (e.g., fuel) and ordinary cargo (e.g., milk).

- These types of containers should implement the IHazardNotifier interface
  - This interface allows for sending a text notification during the occurrence of a hazardous situation along with information about the container number.
- At the moment of initiating the loading method for goods into the container, we should:
  - If the container stores hazardous cargo - it can only be filled to 50% of its capacity
  - Otherwise, it can be filled up to 90% of its capacity
  - If we violate any of the described rules - we should report the attempt to perform a dangerous operation.

Gas containers (G)

Containers storing gas carry additional information about the pressure (in atmospheres).

- When emptying a gas container - we leave 5% of its cargo inside the container.
- It should implement the IHazardNotifier interface. The method should allow for informing about the occurrence of a hazardous event along with the serial number of the container.
- If the mass of the cargo exceeds the allowable payload - we want to return an error.

Refrigerated container (C)

A refrigerated container contains information about:

- The type of product that can be stored in the container.
- The temperature maintained in the container.
- The container can only store products of the same type.
- The temperature of the container cannot be lower than the temperature required by a given type of product.

Example of possible products and temperatures.

Product	Temperature
Bananas	13,3
Chocolate	18
Fish	2
Meat	-15
Ice cream	-18
Frozen pizza	-30
Cheese	7,2
Sausages	5
Butter	20,5
Eggs	19

Our application should allow for the preparation of a given container ship for a voyage. About the container ship itself, we would like to remember:

- All the containers that the ship transports
- The maximum speed the container ship can develop (in knots)
- The maximum number of containers that can be transported
- The maximum weight of all containers that can be transported by the ship (in tons)

We want the application to support the following operations:

- Create a container of a given type
- Load cargo into a given container
- Load a container onto a ship
- Load a list of containers onto a ship
- Remove a container from the ship
- Unload a container
- Replace a container on the ship with a given number with another container
- The possibility of transferring a container between two ships
- Print information about a given container
- Print information about a given ship and its cargo

Then, in the Main method, try to use the classes and methods you have prepared. Check if you are able to perform all the actions described in the text.

Task extension - simulation of application operation (non-compulsory)

For volunteers. Try to prepare a console interface that would allow for the implementation of all functions. An example of the interface operation is shown below.

The user launches the application. The system displays:

mathematicaCopy code

```
List of container ships: None
List of containers: None

Possible actions:
1. Add a container ship
```

The user selects 1. At this moment, the system asks sequentially for all the necessary data. After finishing, the system displays the main screen again.

```
List of container ships: Ship 1 (speed=10, maxContainerNum=100, maxWeight=40000)

List of containers: None

Possible actions:

1. Add a container ship
2. Remove a container ship
3. Add a container
```

After adding a container, it appears on the list of containers. Then the user has the opportunity to place the container on the ship, remove a given container...