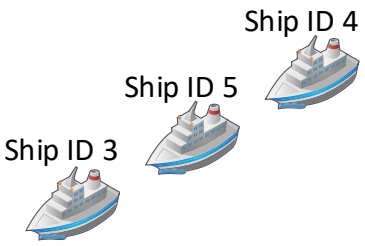
Shipping

You finally achieved your dream of managing Omazan's large convoy of ships; it's so futuristic it's next gen shipping. These ships carry good across the globe. Ships will send in requests to join and leave your convoy, and traders will query the contents of the convoy (either the goods on ships and potentially the ids of said ships). Your job is to write program which stores all this information and services the traders' strange inquiries.

As an example you could have the following set of ships,



Ship 3 Contents
Cocoa, Sugar, Sugar, Grain

Ship 4 Contents
Corn, Carrot

Ship 5 Contents
Car, Car, Car, Car, Car, Car

Ship 3 being the first ship in the convoy. To prevent nautical accidents from happening, you only allow ships to leave when they are the last ship in the convoy. If ship 3 requests to leave (note it's not the last ship in the convoy), it won't be able to immediately. For this reason when Ship 3 requests to leave it will swap positions with ship 4, and then depart from the convoy.

A trader does not know the ids of the ships, so they might request the id of a ship at a location in the convoy. Alternatively, a trader may request the name of an item in some particular container of a specified ship (by location starting index at 1). Suppose we have the original convoy of ships 3, 5, and 4. Imagine a trader asks what the 1^{st} ship's 4^{th} good is. The 1^{st} ship has Ship ID 3. The 4^{th} good is Grain. For this reason Grain would be reported to the trader. If a trader asks for a ship or container not in the convoy, you should report that the ship-container request is not valid.

Input Specification
Each line of input describes either an event or contains the integer -1.

If a ship wishes to leave, the event begins with the integer 0 (zero). Following the integer 0, the id of the ship (1 indexed) will be given. The id given will be a valid ship in the convoy.

If a ship wishes to join the convoy, the event begins with the integer 1 (one). Following the integer 1, a positive integer, ***n***, representing the number of containers on the ship will be given. Following ***n*** will be ***n*** (***n*** $\leq$ 10,000) space separated container descriptions. Each container description will be composed of two values a positive integer ***k*** (k $\leq$ 100) representing the number of alphanumeric characters in the containers name and a string composed of only ***k*** alphanumeric characters.

If a trader requests the id of a ship in some position of the Convoy, the event begins with the integer 2 (two). Following the integer 2, the position of the ship in the convoy will be given. You will need to output the id for such a request.

If a trader requests the contents of some container, the event begins with the integer 3 (three). Following the integer 3, the position of the ship in the convoy will be given. Following the ship position will be an integer representing the container position requested.

If the request in the input is just the value -1 then the program should exit with a successful return code.

Output Specification

For each trader event (type 2 and type 3) you will need to output a line with some value.

In type 2 you will output the id of the ship at the given position. If the ship position is invalid output a -1 instead.

In type 3 you will output the name of the contents of a container at the given position. If the ship/container position is invalid output a -1 instead.

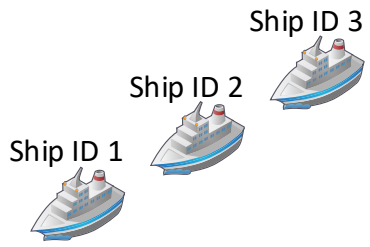
Input Output Example

Input	Output
1 1 3 Dog 1 2 5 AShip 11 AnotherShip 1 4 5 Cocoa 5 Sugar 5 Sugar 5 Grain 0 1 3 2 2 1 2 4 Corn 6 Carrot 2 1 2 2 2 3 1 6 3 Car 3 Car 3 Car 3 Car 3 Car 3 Car 2 2 0 2 2 2 3 3 1 3 1 4 3 4 1 -1	AnotherShip 3 2 4 2 5 Corn Grain -1
1 1 3 DOG 1 1 3 CAT 1 5 1 A 1 B 1 C 1 D 1 E 3 1 1 3 3 1 3 3 5 3 1 5 0 2 2 2 -1	DOG A E -1 3

Explanation

Case 1

We add 3 ships with no trader queries



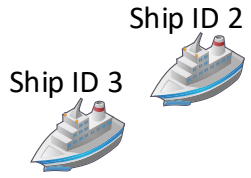
Ship 1 Contents
Dog

Ship 2 Contents
AShip, AnotherShip

Ship 3 Contents
Cocoa, Sugar, Sugar, Grain

Output

We remove the ship with ID 1 by swapping it with ship 3



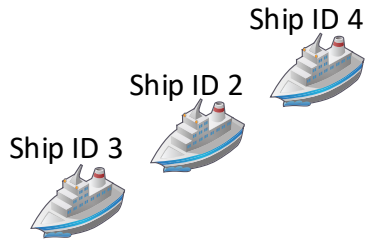
Ship 2 Contents
AShip, AnotherShip

Ship 3 Contents
Cocoa, Sugar, Sugar, Grain

We then query the 2nd ship's 2nd container's contents. Ship 2 happens to be the second ship, (and confusingly enough) the contents are "AnotherShip" quotes for clarity.

Output
`AnotherShip`

We then add a 4th ship



Ship 2 Contents
AShip, AnotherShip

Ship 3 Contents
Cocoa, Sugar, Sugar, Grain

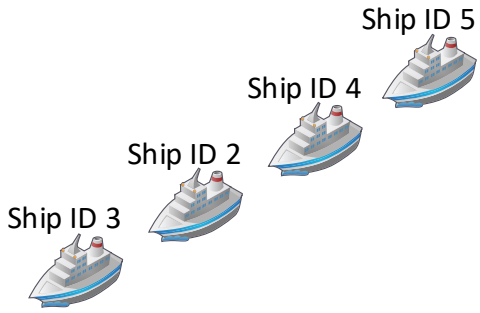
Ship 4 Contents
Corn, Carrot

We then request the ID for each ship from front to back and get the following output

Output

`AnotherShip`
`3`
`2`
`4`

We then add a 5th ship



Ship 2 Contents
AShip, AnotherShip

Ship 3 Contents
Cocoa, Sugar, Sugar, Grain

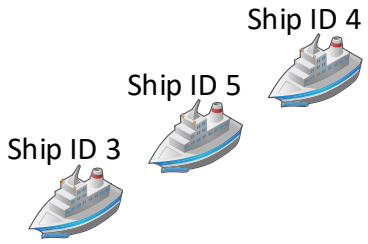
Ship 4 Contents
Corn, Carrot

Ship 5 Contents
Car, Car, Car, Car, Car, Car

We request the ID of the 2nd ship which is still 2.

Output
AnotherShip 3 2 4 2

It appears that 2 had enough of our ship shenanigans, so it leaves, and we have the following Convoy



Ship 3 Contents
Cocoa, Sugar, Sugar, Grain

Ship 4 Contents
Corn, Carrot

Ship 5 Contents
Car, Car, Car, Car, Car, Car

A trader then request the ID for the second ship, which is now 5

Output
AnotherShip 3 2 4 2 5

The contents for the 3rd ship's 1st cargo, the contents for the 1st ship's 4th cargo, and the contents for the 4th ship's 1st cargo are requested. These values are Corn, Grain, and -1 respecitvely

Thus the resulting output is

Output
AnotherShip 3 2 4 2 5 Corn Grain -1

No more queries are given and the program exits

Grading Information

Reading from standard input – 10 points

Writing to standard output – 10 points

Using structures to store the ships – 10 points

No output aside from the answer (e.g. no input prompts) – 10 points

Creates function to add and remove ships – 10 points

Your program will be tested on 10 test cases – 5 points each

No points will be awarded to programs that do not compile.

Solutions without dynamic memory will receive a maximum of 50 points

Only cases that finish within the maximum of {5 times the judge solution, 10 seconds} will be graded.

You will most likely need no memory leaks to successfully complete each case