RUBBER DUCK RACE - RDR

In US (who else) they invented Rubber Duck Races. A lot of YouTube videos shows it, f.ex. a little and a huge race :

https://www.youtube.com/watch?v=6GF6U_CsfBc

https://www.youtube.com/watch?v=NlUm_kBCW9g

They throw an amount of numbered yellow rubber ducks in streaming water.

The owner of the first duck, which crosses the finishing line, wins very often a major prize. You pay to participate and it is possible to play upon the ducks too. The money usually goes to charity.

TECHNICAL HINT

Your program should use queues in the solution. And what is a queue?

As Reges writes :

> Collection
>
> An object which stores a group of other objects, called its elements

Queue

is a collection with FiFo, First In First Out.

Just as a row before a ticket office. The first one to enter the queue should be the first to get a ticket.

EXERCISE 1

Design your own Queue class. It must at least :

1. have a variable number of elements,
2. be able to add a new object in the end
3. remove one from the front of the queue.
4. get the number of objects in the queue.

You are welcome to add other useful features, which takes care of "what if" situations.

Make the design so it is possible for any computer scientist to implement your idea/plan

EXERCISE 2

Design the program needed for The Rubber Duck Race simulation described below.

You can assume the use of the above-mentioned Queue class.

And this design should make it possible for any computer scientist to implement your idea/plan too.

RUBBER DUCK RACE SIMULATION

The Rubber Duck Race is simulated with for - each time step – a number N of queues. They do each have room for up to same number N of unique numbered yellow rubber ducks.

We begin with ten queues of ten yellow rubber ducks, i.e. there are 100 numbered ducks (1 – 100) in our race.

For each time step in our simulation, there 1 less queue with one less room. I.e. in time step number 2, there are only room for the first 81 ducks from time step number 1. In time step number 3 only 64 ducks and so on.

To go from one time step to the next, we draw a random number from 1 – N, and the foremost rubber duck from that queue (if any are left) is transferred via a new random number 1 – (N-1) to the next time steps set of queues.

In time step number 10, there will only be one duck left (1 queue of 1 Rubber Duck) – our winner.