Use Kotlin, C++, Java, TypeScript, or any other object oriented programming language of your choice to complete the following tasks

A. Build Queuing classes

a. Write 2 classes to implement the following `IQueuable` interface

```
Interface IQueuable {
//adds value to queue and returns new queue
enqueue(value: string): string[];

//removes item from queue, and returns the item removed
dequeue(): string;

//returns a list of all the items in the queue
getQueue(): string[];

//returns the number of items in the queue
size():number;
}
```

Build your queues ontop of arrays; call your classes `FIFOQUEUE` and `LIFOQUEUE` (first-in first-out & last-in first-out) queues, or `QUEUE` and `STACK`, whichever names you prefer.

b. If you've written your enqueue and dequeue using array methods, how would you rewrite the same functions without using any array methods?

c. Show how you would improve the design of your classes, apply various design patterns and techniques as you see fit

B. Build a simple carwash simulating mobile app.

Imagine carwash where customers drive in, park their cars, get a unique ticket number for their car. The carwash business has a strict first-come first-serve policy, and each car is tagged with a copy of the ticket number for easy identification

Write a very basic mobile application to simulate the process

- A customer can drop of their car and get a unique ticket number
- A customer can collect their car only if it's been washed. They do so by entering their ticket number, and
    - If the car has been washed, a thank you message is displayed
    - If the car is still in the queue, or is being washed, an appropriate message is displayed
- You can create a WASH task as a simple timeout operation the runs for a pre-determined amount of time. Only one car can be washed at a time.
- User should be able to view the queue, which would show which cars are in waiting, which car is currently being washed, and which cars have already been washed but not picked up

You will score more points if you make use of the code written in part A above.

Note:

1. Host your source code on any public repository for easy sharing.
2. You may use a simulator or actually create an installable app to demonstrate your work