# Kopernikus Rust-FFI code challenge

## Setup

This challenge was tested using Rust v1.61 and GCC v11.2 on 64-bit Linux, but that specific setup isn't required.
Please let us know if you find any operating system/hardware related issues.

## Your tasks

Write some Rust code that interacts with the given C code.

1. Define *VehiclePose* in Rust, in a way that's compatible with the definition in the *kpns.c* file.
2. Call the *create* C function and display the vehicle state.
3. Create 10 vehicles, put them in a vector and call the *translate* function.
4. Access and print the *next_id* variable defined in the C code.
5. Call the *create_heap* function and convert the resulting nullable pointer into an Option containing a reference.
6. Put the Rust definitions of the C functions in a module called *ffi* and remove the need to use *unsafe* when calling the functions.

Bonus:

- 1. Compile the C code as a shared library and load it in Rust dynamically. To compile it as a shared library on Linux, you can use gcc -shared kpns.c -o libkpns.so
- 2. Spawn two threads and pass the reference to the heap allocated VehiclePose from one thread to the other.

The idea here is that we get a better idea of your current Rust skills.
Looking at documentation is fine, but please don’t glue together answers from online forums without understanding what you’re doing.