RUST Launch Pad

Worksheet # 2

1. Define a struct called `Person` with fields for `name` (String) and `age` (u32). Create an instance of the `Person` struct and print its `name` field.
2. Define an enum called `Color` with variants for Red, Green, and Blue. Write a function that takes a `Color` enum as an argument and returns a corresponding RGB value as a tuple.
3. Write a function that accepts a tuple `(i32, i32)` and returns the sum of its elements.
4. Create an Option enum to represent either a string or a number. Write a function to print the value if it's a number.
5. Write a function that borrows a string and appends " World!" to it.
6. Define a struct `Book` with a title (String) and implement a method `get_title` that returns a reference to the title.
7. Define a struct named `Book` with fields `title`, `author`, and `pages`.
   a. Create an instance of the `Book` struct and print its title.
   b. Define an enum named `Status` with variants `Active`, `Inactive`, and `Suspended`.
   c. Write a function that takes a `Book` and a `Status` and returns a tuple containing the book's title and its status.
8. Given the enum `Status` from the previous section, use pattern matching to write a function that returns a string description of each variant.
   a. Destructure the `Book` struct to extract and print the author's name.
   b. Write a match statement for an `Option<i32>` that prints "Has a value" if it's `Some` and "No value" if it's `None`.
9. Create a function that takes ownership of a `Book` and returns its title.
   a. Write a function that borrows a `Book` and modifies its title. Ensure it doesn't consume the book.
10. Create a module named `utils` and place a function inside it named `display_book` that prints a book's details.
    a. Use the `pub` keyword to make the `Book` struct and its fields public.
    b. Import the `utils` module in another module and use the `display_book` function.
11. Create a new library crate named `bookshelf`.
    a. Add the `Book` struct and `Status` enum to the library.

    b. Use the `bookshelf` library in another Rust project and create an instance of the `Book` struct.

12. Initialize a new Git repository in your Rust project.
    a. Commit your changes with the message "Initial commit".
    b. Add a dependency from `crates.io` to your `Cargo.toml` file. (You can choose any crate for this task.)
    c. Update your project's dependencies using Cargo.
    d. Publish a new version of your `bookshelf` library to `crates.io`. (Note: This is a theoretical task, as you shouldn't actually publish without a real library.)