Universität Bielefeld
Technische Fakultät

AG Genominformatik
Dr. Lucas Robidou

The Rust programming language
Summer 2024 / 2025

Exercises

1 Rust macros

1. Write a Rust macro that takes two numbers, prints them, and returns their sum.
2. Write a Rust macro that, given a name, a type, and a value, declares a variable of that name, with that type, and initializes it to that value.
3. (advanced) Write a Rust macro that takes an arbitrary number of numbers and returns them normalized, so that their sum is 1. You can return e.g. a Vec.

   ```
   pub fn main() {
       println!("{:?}", normalize!(1, 2, 3)); // Output: [0.166..., 0.333..., 0.5]
   }
   ```

4. (advanced) Make the macro return a tuple instead of a Vec.

You can write a macro that takes one or two arguments like this:

```
macro_rules! log {
    ($val:expr) => { $val.log(2.0) };  // no base provided: assuming it's 2
    ($val:expr, $base:expr) => { $val.log($base) };  // using the base provided
}

pub fn main() {
    let value: f32 = 4.0;
    println!("{}", log!(value)); // 2

    let value: f32 = 1000.0;
    let base: f32 = 10.0;
    println!("{}", log!(value, base));  // 3
}
```

5. (advanced) Write a macro that takes an arbitrary number of numbers and returns their minimum. Hint: make it recursive.