Rust for Clinical Study Reports and Submission

Author One Author Two

Table of contents

Welcome

Welcome to Rust for Clinical Study Reports and Submission.

Preface

In this book

To be added. Knuth (1984)

```
// A trait which implements the print marker: `{:?}`.
use std::fmt::Debug;

trait HasArea {
    fn area(&self) -> f64;
}

impl HasArea for Rectangle {
    fn area(&self) -> f64 { self.length * self.height }
}

#[derive(Debug)]
struct Rectangle { length: f64, height: f64 }
#[allow(dead_code)]
struct Triangle  { length: f64, height: f64 }

// The generic `T` must implement `Debug`. Regardless
// of the type, this will work properly.
fn print_debug<T: Debug>(t: &T) {
    println!("{:?}", t);
}

// `T` must implement `HasArea`. Any type which meets
// the bound can access `HasArea`'s function `area`.
fn area<T: HasArea>(t: &T) -> f64 { t.area() }

fn main() {
    let rectangle = Rectangle { length: 3.0, height: 4.0 };
    let _triangle = Triangle  { length: 3.0, height: 4.0 };
```

```
    print_debug(&rectangle);
    println!("Area: {}", area(&rectangle));

    //print_debug(&_triangle);
    //println!("Area: {}", area(&_triangle));
    // ^ TODO: Try uncommenting these.
    // | Error: Does not implement either `Debug` or `HasArea`.
}
```

References

Knuth, Donald E. 1984. “Literate Programming.” *Comput. J.* 27 (2): 97–111.