Second Last Week

uocsclub

February 19, 2022

Contents

1 Enums

Enums in Rust are a way to reresent data which can take many varied forms. An enum lets you define types by specifying its variants.

To give an intution for what this means and why it is useful, let's talk about the difference bettween enums and structs.

The two fundamental ways to bring together data is a product type and a sum type. A product type gives you a way to specify one type AND another AND another, while a sum type gives you a way to bring together one kind of data OR another OR another. In the language of type theory (and mathematics), ANDing is a product ($\times$) type, while ORing is a sum ($+$) type.

You can combine these in interesting ways to fit your data to your particular tasks.

For example, if you wanted to create the type representing ((a string AND an int) OR (a bool AND an int)), you would express this as (String $\times$ int) + (Bool $\times$ int)

1.1 Defining Enums

In rust, to define an enum, we use the `enum` keyword, and then define it as you might define a struct, listing its variants.

```
enum IP {
    V4,
```

```
    V6,
}
```

In mathematics, we would express this as there being two unit (meaning they only take on one value, the general "inhabitation" of the type) types V_4 and V_6, and we define $IP := V_4 \times V_6$.

When we then want to use this enum, we can select the variant using the method syntax:

```
let i1 = IP::V4;
let i2 = IP::V6;
```