Rust 言語のためのマクロ機能の実現に向けて

杉山俊　201903016

2023 年 1 月 30 日

1　はじめに

Rust 言語は、c 言語、c++ に代わるプログラミング言語を目指しているプログラミング言語である。rust 言語はマルチパラダイムプログラミング言語であり、手続きプログラミング、オブジェクト指向プログラミング、関数型プログラミングなどの実装手法をサポートしている。
現在、rust 言語が抱える問題点の一つとして、学習難度が高い言語とされているというものがある。その原因は、rust 言語は c++ の問題点を解消するためにプログラマーが開発した言語であるため、プログラミング初心者にとっては難しい独自の概念が導入されていることと、利用人口が少ないため情報が多く出回っていないということがある。そこで本研究では rust のマクロ機能を用いてコードを書くことの簡略化と自分があったら便利と感じたものを追加していくことを目的として進めていく。

2　実装方法

rust 言語のマクロには、大きく分けて宣言的マクロと、手続き的マクロの 2 つに分類される。宣言的マクロは match 式と似たようなものをかけるマクロで、match 式とは式を受け取り、式の結果の値をパターンと比較し、マッチしたパターンに紐づいたコードを実行するものである。宣言的マクロでは受け取るものが rust のソースコードそのものを受け取り、パターンがソースコードの構造と比較されるというものである。手続き的マクロは、コードを入力として受け取り、そのコードに対して作用し、出力としてコードを生成するものである。この研究ではコードを書くことの簡略化や機能追加のために引数の数や型が違い、それぞれ実行される機能も異なるため、型の種類や数が決まってしまう手続き的マクロではなく、パターンマッチングが行われ、様々な状況に対応することができる宣言的マクロを使用していくこととする。宣言的マクロの実装手順は
「1」macro __ rules!と書き、その後、定義したいマクロ名を付ける。
「2」受け取る引数の型と個数を決める。型が異なり分岐する場合はかっこを閉じた後新しい型を書く。各グループセミコロンで終わる必要がある。
「3」引数に応じて動かしたいプログラムを記入する。
となっており、引数が分岐の条件になるパターンマッチングのような見た目と分かりやすくなっている。
実装例

```
macro_rules! sotu {
  (kei => $x:tt,$y:expr,$($z:expr),* ) =>
  { if $x == "+" { //x が + ならば
      let mut kei = $y; //kei に y をセット
      $(
       kei = kei + $z; //足す
      )*
      println!("{}",kei) //結果表示
    } else if ~~ //差積商も同様
  };
  (len => $x:tt,$y:tt,$z:tt) =>
    {if $x == "len" {
      println!("{}{}",$y,$z)
     }};
```

```
}
```

3　現在の課題

「1」四則演算のうち 1 つずつしか使うことができず、2 つ入れようとすると組み合わせが大量に増えてしまいコードの量が普通に書いた方がはやくなってしまう。
「2」計算結果、文字列を出力するだけで x=6 のように変数にマクロ内で操作したものをもとのコードに適用することができていないため汎用性が低くなってしまっている。
「3」今回のマクロは最初の引数にキーワードをいれることで条件分けをしたため、当然のことながらマクロを使用するときにはキーワードを知っておく必要がある。
「4」コードの簡略化はできたものの、rust 独自の概念をこのマクロを使って表現するのは難しかったためほかの方法を試す必要がある。

4　まとめ

今回の研究では宣言的マクロを使うことによって計算や文字列を連結させる機能を追加することができた。パターンマッチングと似た形であったため単純な作業を省くことには実現できたものの複雑な機能を作ることができなかったので、特定の 1 つの機能をもつマクロを作る際には自由度の高い手続き的マクロを用いることでより多くの機能に対応し作ることができるようになるだろう。

参考文献

[1] The Rust Programming Language.
`https://doc.rust-jp.rs/book-ja/`

[2] Rust by Example 日本語版.
`https://doc.rust-jp.rs/rust-by-example-ja/`

[3] プログラミング言語 Rust のススメ.
`https://qiita.com/k5n/items/758111b12740600cc58f`