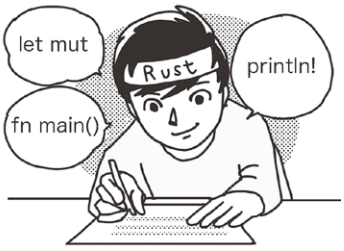
Rustプログラミング問題集

ダウンロード・データあります



第2回　簡単なアルゴリズムを実装する

中村 仁昭，懸川 岳

本連載ではRustを使った問題とその回答でRustを学んでいきます．回答のプログラムはRustでよく使う機能を選んでいるので，Rustのプログラム集にもなっています．プログラムを作るときのコード・スニペットにもなるでしょう．

今回は代表的なアルゴリズムをRustで実装してみます．（編集部）

● 実行環境

rustupでインストールした標準的な環境で動作確認しています．rustupでのインストール手順は公式ページ（https://www.rust-lang.org/ja/tools/install）を参照してください．インストールが難しい場合はウェブ・ブラウザの実行環境Rust Playground（https://play.rust-lang.org/）でも試せます．

1 素数

2から100までの素数を表示してください．

● 回答

回答を**リスト1**に示します．forループなどで範囲指定をする際に，begin..endで指定するとbeginは含みますがendは含まないことに気を付けてください．endを含めたい場合にはbegin..=endのような記述が可能です．

リスト1　素数を表示する

```
fn is_prime(n: u32) -> bool {
    if n <= 1 {
        return false;
    }

    for i in 2..=(n as f32).sqrt() as u32 {
        if n % i == 0 {
            return false;
        }
    }
    true
}

fn main() {
    for i in 2..=100 {
        if is_prime(i)  {
            println!("{} is prime", i);
        }
    }
}
```

2 ソート

次の数列をクイック・ソートで昇順にソートしてください．

```
[15, 5, 33, 42, 9, 18, 23, 16, 12, 2, 51]
```

● 回答

回答を**リスト2**に示します．一般的なクイック・ソートです．必要な箇所のmutのつけ忘れに注意しましょう．

リスト2　クイック・ソートを行う

```
fn quick_sort<T: Ord + Copy>(array: &mut[T]) {
    if array.len() < 2 {
        return;
    }

    let pivot = array[array.len() / 2];
    let mut i = 0;
    let mut j = array.len() - 1;

    loop {
        while pivot > array[i] {
            i += 1;
        }
        while pivot < array[j] {
            j -= 1;
        }
        if i >= j {
            break;
        }
        array.swap(i, j);
        i += 1;
        j -= 1;
    }
    if i > 0 {
        quick_sort(&mut array[.. i]);
    }
    if j < array.len() - 1 {
        quick_sort(&mut array[j + 1 ..]);
    }
}

fn main() {
    let mut array = [4, 6, 2, 9, 3, 1, 3, 5, 6];
    quick_sort(&mut array);
    println!("{:?}", array);
}
```