08-cutr

目标：实现 cut 命令

实现

- 解析正整数

Rust

```rust
fn parse_index(input: &str) -> Result<usize, String> {
    let value_error = || format!("illegal list value: \"{}\"", input);
    input
        .starts_with('+')
        .then(|| Err(value_error()))
        .unwrap_or_else(|| {
            input
                .parse::<NonZeroUsize>()
                .map(|n| usize::from(n) - 1)
                .map_err(|_| value_error())
        })
}
```

- 创建一个不带参数的格式化错误的闭包
- Use `str::parse` to parse the input value, and use the turbofan to indicate the return type of `std::num::NonZeroUsize`, which is a positive integer value.（指定类型解析）
- If the input value parses successfully, cast the value to a usize and decrement the value to a zero-based offset.
- If the value does not parse, generate an error by calling the value_error closure

- 解析区间

Rust

```rust
fn parse_pos(range: &str) -> MyResult<PositionList> {
    let range_re =  Regex::new(r"^(\d+)-(\d+)$").unwrap();
    range
        .split(',')
        .into_iter()
        .map(|val| {
            parse_index(val).map(|n| n..n + 1).or_else(|e| {
                range_re.captures(val).ok_or(e).and_then(|captures| {
                    let n1 = parse_index(&captures[1])?;
                    let n2 = parse_index(&captures[2])?;
                    if n1 >= n2 {
                        return Err(format!("First number in range ({}) must be lower than second number ({})", n1+1, n2+1));
                    }
                    Ok(n1..n2 + 1)
                })
            })
        })
        .collect::<Result<_, _>>()
        .map_err(From::from)
}
```

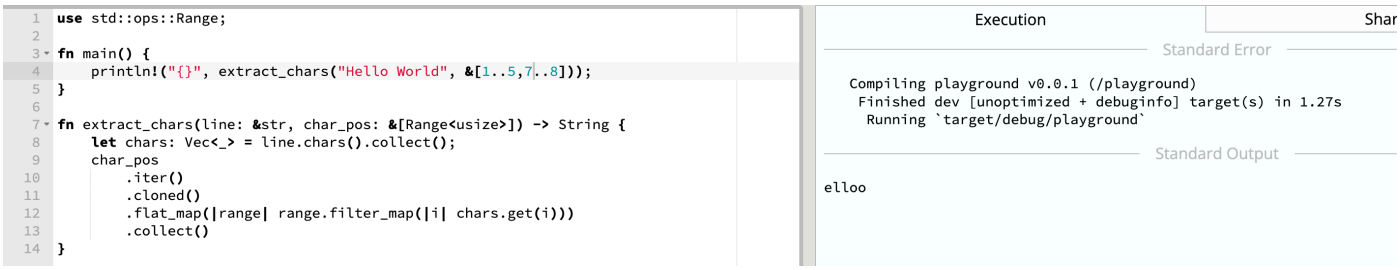
- Create a regular expression to match two integers separated by a dash, using parentheses to capture the matched numbers.（创建一个两个数字用 ‘-’ 分割的正则表达式）
- Split the provided range value on the comma and turn the result into an iterator. In the event there are no commas, the provided value itself will be used.（用 ‘,’ 分割开，转成迭代器）
- Map each split value into the closure.
- If `parse_index` parses a single number, then create a Range for the value. Otherwise, note the error value e and continue trying to parse a range.
- If the Regex matches the value, the numbers in parentheses will be available through `Regex::captures`.
- Parse the two captured numbers as index values.
- If the first value is greater than or equal to the second, return an error.（第一个数比第二个数大报错）
- Otherwise, create a Range from the lower number to the higher number, adding 1 to ensure the upper number is included.
- Use `Iterator::collect` to gather the values as a Result.
- Map any problems through `From::from` to create an error.

- 取出每一行的区间值

Rust

```rust
fn extract_chars(line: &str, char_pos: &[Range<usize>]) -> String {
    let chars: Vec<_> = line.chars().collect();
    char_pos
        .iter()
        .cloned()
        .flat_map(|range| range.filter_map(|i| chars.get(i)))
        .collect()
}
```

- 可以看下面的例子



https://play.rust-lang.org/?version=stable&mode=debug&edition=2021&gist=dac384e59501f214301c52e0ba936ac8