answer

- `\w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*` 正则表达式匹配的字符串的含义是什么？

该正则表示式是匹配的邮箱字符串

- \w+表示一个或多个字母、数字或者下划线
- ([-+.]\w+)*表示- + .和字母数字下划线组合0次或多次
- @为邮箱固定符号
- \w+同上
- ([-.]\w+)*表示- .和字母数字下划线组合0次或多次
- \ .为固定符号
- \w+同上
- ([-.]\w+)*表示- .和字母数字下划线组合0次或多次

例如112233@ustc.edu.cn

- 匹配 HTML 注释：编写一个正则表达式，可以匹配 HTML 中的注释，例如 `<!-- This is a comment -->` 。

<!--(.*?)-->使用非贪婪匹配每次只匹配一次，方便分别匹配多个注释，若使用(.*)则无法区分多个注释，只能匹配为一整个注释

> 如果存在同时以下规则和动作，对于字符串 `+=`，哪条规则会被触发，并尝试解释理由
>
> ```
> %%
> \+ { return ADD; }
> = { return ASSIGN; }
> \+= { return ASSIGNADD; }
> %%
> ```

return ASSIGNADD.

选择去匹配匹配文本最多的规则

> 如果存在同时以下规则和动作，对于字符串 `ABC`，哪条规则会被触发，并尝试解释理由
>
> ```
> %%
> ABC { return 1; }
> [a-zA-Z]+ {return 2; }
> %%
> ```

return 1

匹配长度相同优先先匹配成功的

> 如果存在同时以下规则和动作，对于字符串 `ABC`，哪条规则会被触发，并尝试解释理由。
>
> ```
> %%
> [a-zA-Z]+ {return 2; }
> ABC { return 1; }
> %%
> ```

return 2

同上，匹配长度相同优先先匹配成功的，这里是2先被匹配

> - 上述计算器例子的文法中存在左递归，为什么 `bison` 可以处理？

自顶向下不可以处理左递归，但是bison语法树是自底向上归约的，所以可以处理

> - 能否修改计算器例子的文法，使得它支持除数 0 规避功能？

```
case '/': $$ = $1 / $3; break; // 这里会出什么问题?
```

这一行做一个判断，除数为0返回一个错误提示即可

```
case '/': if($3 ==0) {$$ = 0; printf("error\n");}
        else $$ = $1 / $3; break;
```