lab1

正则表达式

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

  匹配电子邮件地址.

- **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 -->` 。**

```
<!--(.*?)-->
```

Flex

- **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 ASSIGNADD; }` 将被触发　原因：手册中写到"If it finds more than onematch, it takes the one matching the most text "

- **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; }` 将被触发

- **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; }

原因：手册中写到"If it finds two or more matches ofthe same length, the rule listed first in the 'flex' input file is chosen."匹配的字符串相同长度时，优先匹配序列中首次出现的规则。

Bison

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

Bison 能够处理左递归的原因在于，它在内部生成递归下降分析器，该分析器可以递归地展开规则，直到达到一个终结符或无法继续展开为止。它

通过　不断展开规则来识别左递归并生成相应的右递归形式。

- **能否修改计算器例子的文法，使得它支持除数 0 规避功能?**

```
#include<stdlib.h>
...
...
...
...

term
: factor
{
    $$ = $1;
}
| term MULOP factor
{
    switch ($2) {
    case '*': $$ = $1 * $3; break;
    case '/': {
        if($3==0){
            printf("Division by zero!\n");
            exit(1);
        }
        else{
            $$ = $1 / $3;
            break;
        }
    }
    }
}
```