**词法分析器（正则表达式转DFA）**

——111162 燕飞扬

输入正则表达式规则：

1. 或只能用|不能用+
2. 大写E表示空字符，不要作为输入
3. 请输入英文括号

实现思路：

1、首先，给输入的正则表达式添加连接符’+’，添加算法为：

扫描正则表达式字符串，

当遇到非运算符时，需要对其后面的符号进行判断，若遇到左括号或非运算符时，则需 要在此非运算符后添加连接符'+'；

当遇到运算符'\*'或’）’时，需要判断其右边的符号，若右边的符号为’\*’或’）’，则继续向 下看一位，直到不再遇到’\*’和’）’，若最后遇到左括号或非运算符，则添加连接符’+’。

2、然后用一个运算符栈将中缀表达式转后缀表达式，转化算法为:

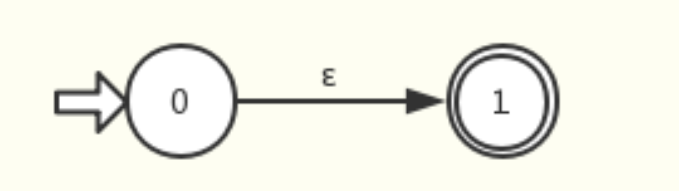
创建一个作为输出的空字符串s，扫描添加了连接符’+’的正则表达式，

当遇到非运算符时，直接追加到s后面

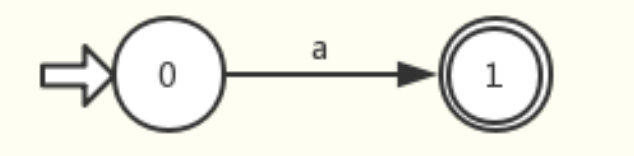
当遇到运算符时，根据运算符的优先级入栈出栈即可，只需保证栈中从栈顶到栈底优先 级依次降低即可（具体优先级在这里不加赘述）。

3、接下来就是由后缀计算出NFA的实现了，NFA的计算用Thompson算法，即：

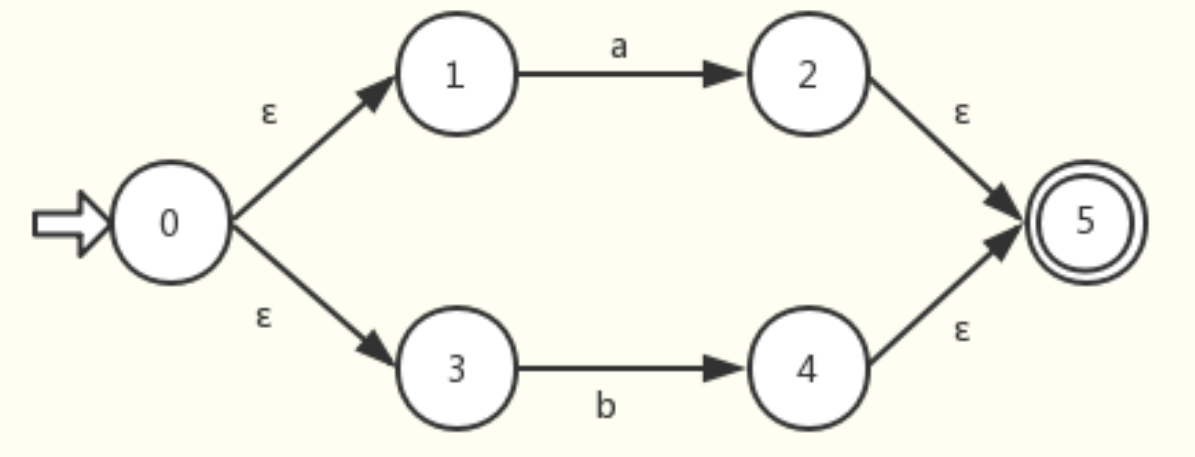
输入为空：



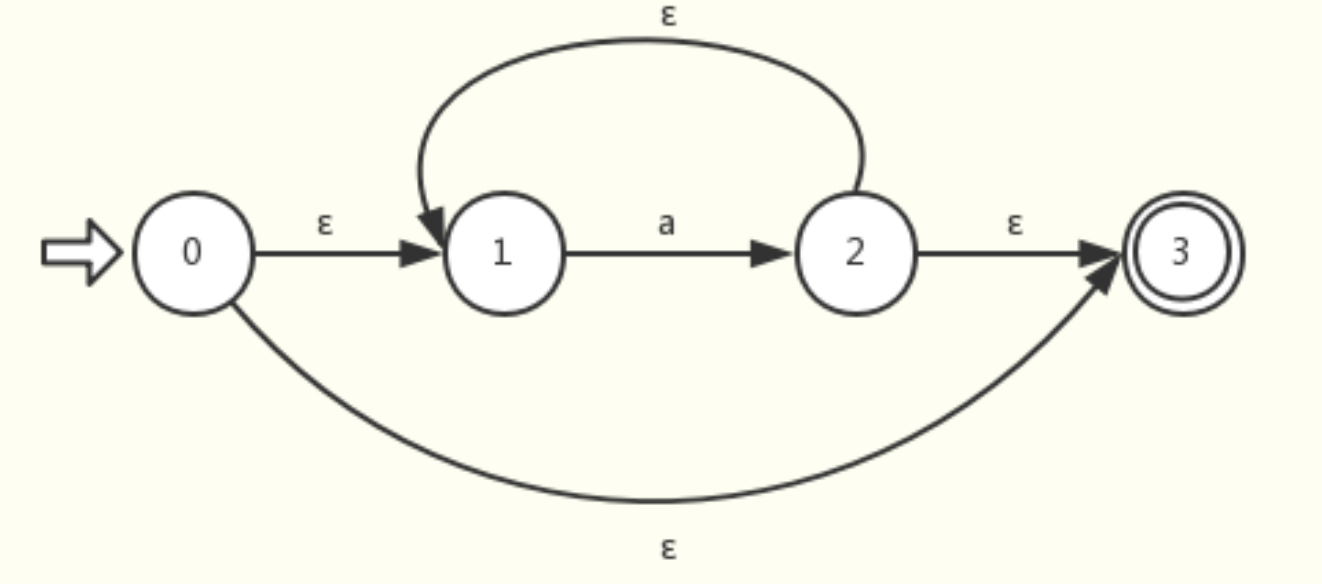
输入为单个字符a



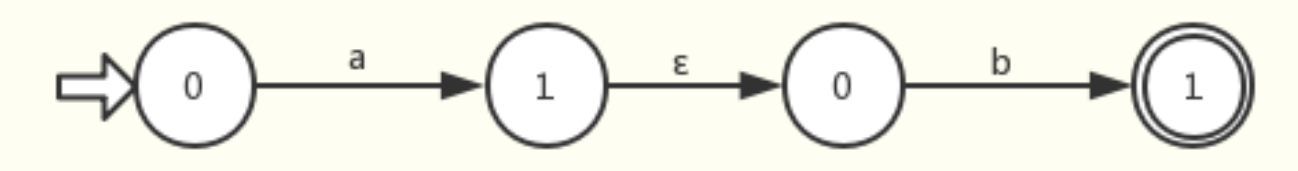
输入a|b



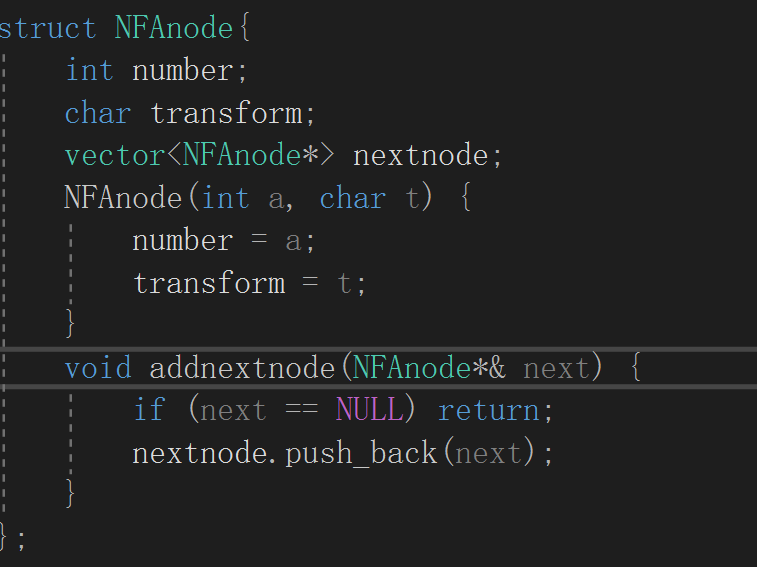
输入a\*



输入ab



NFA的存储方式又是一个问题，看到NFA是一个个结点由指针连接构成，想到了数据结构中的链表，我创建了一个结构体用于表示结点，结构体中vector动态存储子节点指针。



存储问题解决了，接下来就是要重载NFA的运算符’|’，’+’，’\*’,重载运算用Thompson算法实现即可。

下一步就是要计算后缀表达式了，计算方法用一个NFA栈计算即可。

4、NFA转DFA，NFA转DFA的方法用上课讲的建表的方法，

第一点是找到NFA中某个NFA结点的等价结点（由空字符可转换过去的结点）, 只 需从这个结点遍历NFA寻找即可。

第二点是写一个函数实现找出从一个DFA结点经过一个转换字符得到的NFA结点集合， 再判断此集合的等价NFA结点，并加进去。

第三点，模拟建表的不再出现新DFA结束，用一个队列即可实现。

5、DFA最小化

先将DFA分为终态和非终态，然后分别判断DFA状态是否可以区分，将不可区分的分 到一组即可。

6、合并

根据第5步得到的待合并DFA组，将要合并合并为一个即可。

遇到的问题及解决方案：

问题一：遍历NFA的时候采用递归遍历，但是当出现闭包运算的时候会出现环，然后一直循环递归。

解决方案：定义了一个bool型的数组flag，用来指示结点是否被遍历过，当遍历过（flag为true），则结束这条路的递归。

问题二：重载了运算符，当计算后缀表达式的时候，没有考虑出栈的2个NFA的运算顺序，导致出错。

解决方案：NFA的运算不满足交换律，出栈的第一个NFA是右操作符，第二个NFA是左操作符。

问题三：NFA转DFA的时候，建表的过程不知道怎么模拟不再产生新的DFA结束

解决方案：创建了一个DFAnode的队列queue,每次有新结点产生则入队列，当队列为空则说明不再产生新结点。

问题四：调试的时候，输入了中文括号，导致DFA出不来

解决方案：括号应输入英文括号

测试结果截图：

