编号：

**《编译原理》实验报告**

**题 目 ： 词法分析**

**姓 名 ： 徐 凡**

**学 号 ： 202215000141**

**院 （系）： 计算机与人工智能学院**

**指导教师： 张坤丽**

**成 绩 ：**

**2024年11月1日**

1. **实验目的**

**1．设计与实现一个词法分析器，加深对词法分析原理的理解**

**2．掌握将多个NFA合并、将NFA确定化成DFA、将DFA最小化的方法，加深对自动机的理解。**

**3．理解和掌握正则表达式RE到非确定有穷自动机NFA的转换方法，加深对正则表达式和自动机的理解。**

**4．使用Lex工具，设计与实现一个制定语言的词法分析器，加深对词法分析原理的理解，熟练使用Lex。**

1. **问题描述**

**2.1 词法分析器与基于Lex的词法分析器的设计与实现实验内容**

**2.1.1 需要实现的功能**

**输入：源程序字符串，源程序存储在文本文件中（编码格式ANSI），文件名作为命令行参数输入。**

**输出：输出token序列到标准输出设备，并给出该语言词法描述的自动机。**

**待处理符号：**

**2.1.2实验原理和方法**

**1．预处理。实现源程序的输入，从键盘输入流、文件字节流等输入若干行语句，依次存入输入缓冲区（字符型数据），去掉输入串中的回车符、换行符和跳格符等编辑性文字，多个空白符合并为一个。**

**2．词法分析实现。**

**（1）构造二元组：syn，token，sum构成的序列。token用来存放构成单词符号的字符串；sum用来存放整型单词；syn用来存放单词符号的种别编码。**

**（2）设计关键字表：单词符号的种别码(syn)**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **符号** | **种别码** | **符号** | **种别码** | **符号** | **种别码** | **符号** | **种别码** | **符号** | **种别码** | **符号** | **种别码** |
| **main** | **1** | **to** | **10** | **break** | **19** | **“#”** | **28** | **“;”** | **37** | **“>=”** | **46** |
| **if** | **2** | **step** | **11** | **char** | **20** | **“+”** | **29** | **“:”** | **38** | **“==”** | **47** |
| **else** | **3** | **switch** | **12** | **bool** | **21** | **“-”** | **30** | **“{”** | **39** | **“!=”** | **48** |
| **while** | **4** | **case** | **13** | **and** | **22** | **“\*”** | **31** | **“}”** | **40** | **ID** | **49** |
| **do** | **5** | **of** | **14** | **or** | **23** | **“/”** | **32** | **“[”** | **41** | **NUM** | **50** |
| **repeat** | **6** | **default** | **15** | **not** | **24** | **“=”** | **33** | **“]”** | **42** | **Letter** | **51** |
| **until** | **7** | **return** | **16** | **read** | **25** | **“>”** | **34** | **“(”** | **43** | **Digit** | **52** |
| **for** | **8** | **int** | **17** | **write** | **26** | **“<”** | **35** | **“)”** | **44** |  |  |
| **from** | **9** | **real** | **18** | **then** | **27** | **“,”** | **36** | **“<=”** | **45** |  |  |

**2.2 NFA转DFA并最小化实验内容**

**2.2.1需要实现的功能**

**（1）设计一个函数（方法），实现把两个NFA的合并；**

**（2）设计一个函数（方法），实现把NFA确定化成一个DFA；**

**（3）设计一个函数（方法），实现把DFA最小化；**

**（4）输入多个NFA：NFA描述存储在文本文件中，文件名作为命令行参数输入；**

**（5）输出合并、最小化以后的DFA到标准输出设备。**

**2.2.2 实验原理和方法**

**（1）定义类结构：定义NFA和DFA的类结构，分别用于表示非确定性有限自动机（NFA）和表示确定性有限自动机（DFA）。**

**（2）读取与解析NFA：从文件中读取并构造一个NFA对象。**

**（3）NFA合并：将两个NFA合并，形成新的NFA，便于生成复合状态。**

**首先进行两个NFA的合并，步骤如下：**

**1.生成新起始状态：新起始状态编号为两个NFA状态集合中最大值加1。**

**2.合并状态：包括两个NFA的状态集合和新起始状态。**

**3.合并接受状态：直接合并两个NFA的接受状态。**

**4.合并转移函数：通过字典扩展合并两个NFA的转移函数。**

**5.添加空转移：从新起始状态到原两个NFA起始状态添加空转移。最终得到合并后的NFA。**

**若输入多个NFA文件，则重复进行以上步骤。**

**（4）NFA到DFA转换：使用子集构造法将NFA转为DFA。**

**1.初始化DFA状态和转换结构：**

**使用stateSet集合存储元组<转换后的DFA状态，NFA对应的状态集>。使用dfaStates集合存储转换后的DFA状态。transitions用于存储转换规则。**

**2.定义起始状态：DFA的起始状态startState是NFA的起始状态。**

**3.广度优先搜索（BFS）：**

**使用队列queue记录待处理的状态集合。currentState记录当前正在查找的状态集合，先加入startState，寻找其ε-Closure，得到I。**

**将I加到状态集合stateSet和队列queue。若队列不空，则取出队头状态集合，遍历字母表中的每个符号，寻找转换字符为signal的Closure，再寻找其ε-Closure，找到从当前状态集合可以转移到的下一个状态集合。**

**4.更新DFA状态与转移：**

**对于每个状态集合，检查是否出现在总的状态集合和队列中，若未出现，将其加入并赋予编号。在transitions中记录该状态的转换。**

**5.返回构建的DFA：包含所有确定化后的状态和转移规则。**

**（5）DFA最小化：使用分区算法对DFA进行最小化处理。即将状态集合划分为若干个等价类，通过不断细分这些类，最后得到最小化DFA。**

**1.初始划分：将DFA状态分为不包含终态的状态和包含终态的两个分区。**

**partitionMap字典记录每个状态所属的分区编号。**

**2.迭代细化划分：针对每个分区，根据状态转移情况进一步细分。对于每个状态和输入符号，检查其转移到的状态所在的分区，若指向不同的分区，则将它们划分到不同的分区。**

**3.分区稳定性检查：重复以上过程，直至分区不再发生变化，说明最小化划分已完成。**

**4.构建最小化DFA：将每个分区视为最小化后的DFA状态，构建转移函数并确定接受状态和起始状态。**

**5.返回一个最小化的DFA。**

**2.3 RE到NFA的转换实验内容**

**2.3.1需要实现的功能：**

**（1）设计一个函数（方法），实现把RE确定化成一个NFA；**

**（2）正则式RE作为命令行参数输入；**

**（3）输出转换好的NFA到标准输出设备。**

**2.3.2实验原理和方法**

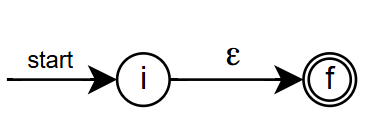
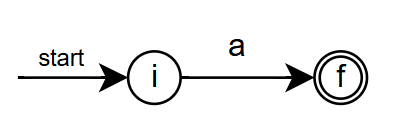
**（1）正规式转换为后缀表达式。对正规式按如下规则添加分隔符‘.’，然后按照双栈法将其转变为后缀表达式，使计算机正确识别输入的正规式。**

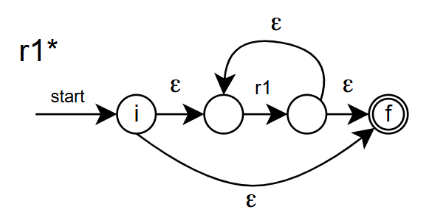
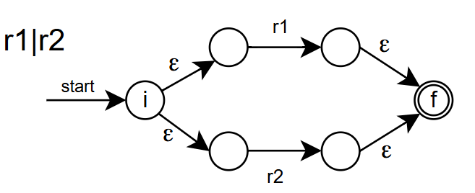
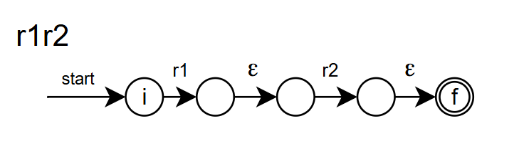
**①若第二个是字母、第一个不是'('、'|'要添加分隔符。**

**②若第二个是'(',第一个不是'|'、'(',也要加分隔符。**

**（2）节点构造采用Thompson算法。原理是先构造识别子表达式的ε-NFA，再通过几个以下几条规则将ε-NFA合并，最终得到识别完整正则表达式的ε-NFA。**

**（3）若R是复合正规式，从左至右依次建立每个符号或子表达式的ε-NFA，，再通过以下规则合并，最终能构造出识别完整正则的ε-NFA。整个分裂过程中，所有新结点均采用不同的名字。**

 ** **

**图 1 R＝Φ，构造NFA如图 图 2 R＝ε，构造NFA如图 图 3 R＝a(a∈∑)，构造NFA如图 **

**图 4转换规则1 图 5转换规则2 图 6转换规则3**

1. **软件设计方法的选择**

**软件设计方法：面向对象设计的方法。**

**开发语言：C++11。开发环境：Visual Stdio，devC++。编译器：g++。**

**编程风格：变量和函数的命名采用小驼峰命名法，第一个单词首字母小写，后面其他单词首字母大写。分隔符（,和;）前一位没有空格，后一位保持空格。花括号均独占一行。操作符左右有空格。**

**创建的模型：对象模型，功能模型**

1. **分析模型**

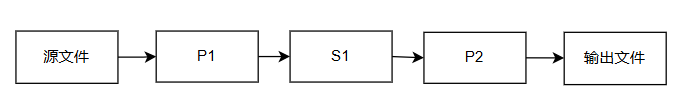
**数据流图：**

**外部实体：源代码文件，输出文件**

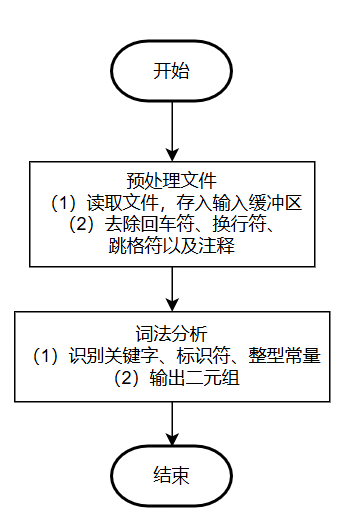
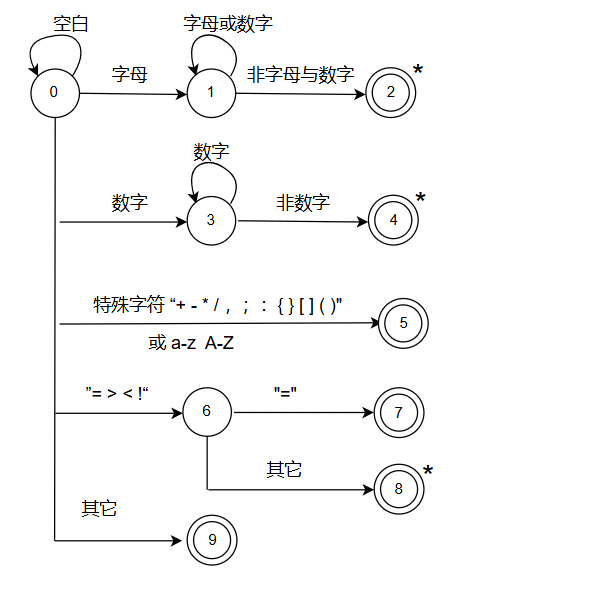
**处理过程：P1预处理（去掉输入串中的回车符、换行符和跳格符移除注释）**

**P2词法分析（识别和输出结果）**

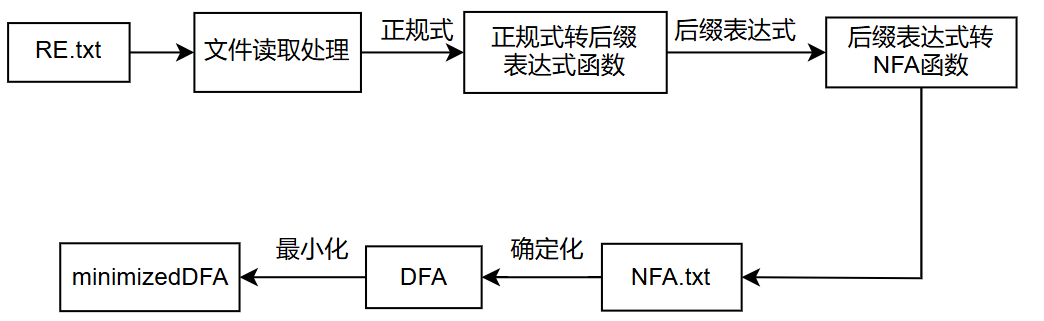
**数据存储：S1预处理后的文件**

****

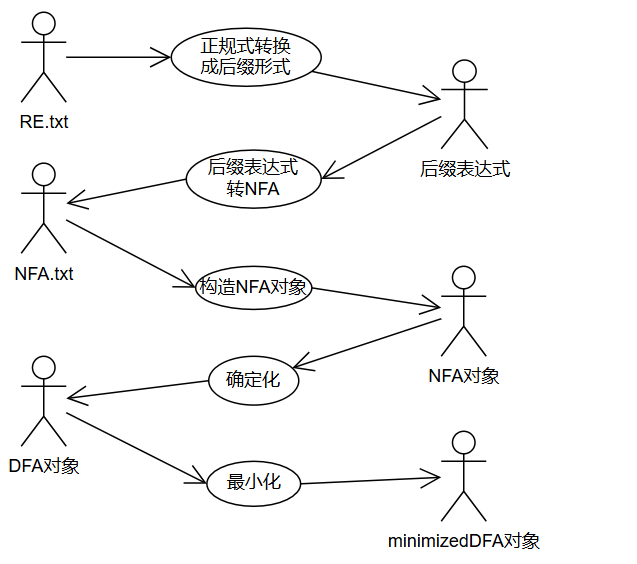
**图 7词法分析器数据流图**

 ****

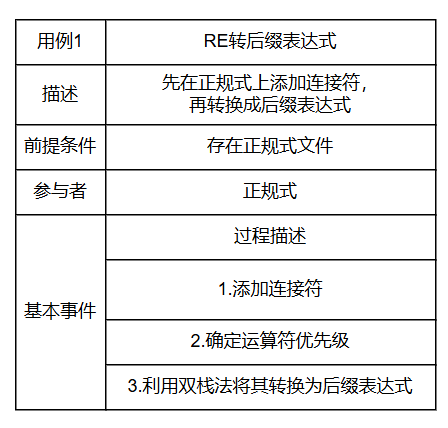
**图 8词法分析流程图 图 9 词法分析器状态转换图**



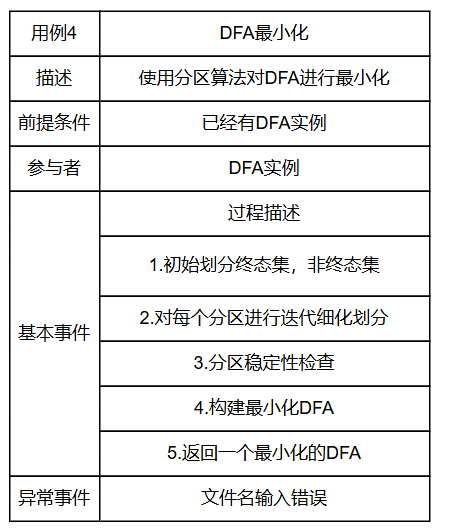
**图 10对象模型**



**图 11用例图**

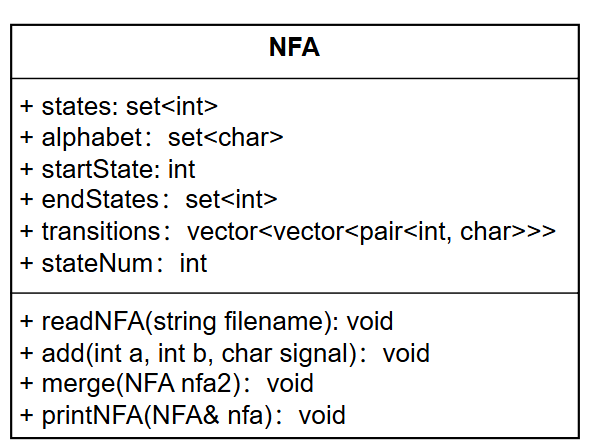
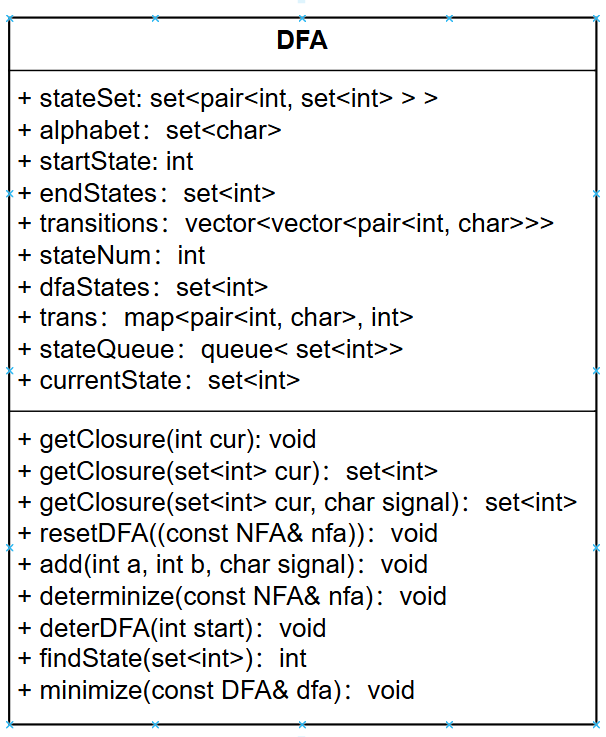
 

**图 12用例文本1 图 13用例文本2**

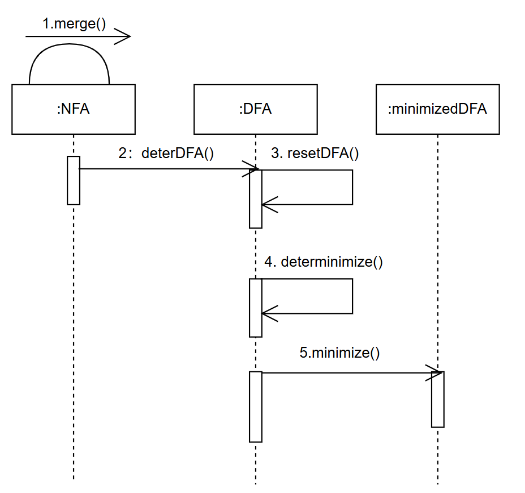
 

**图 14用例文本3 图 15用例文本4**

1. **设计模型**

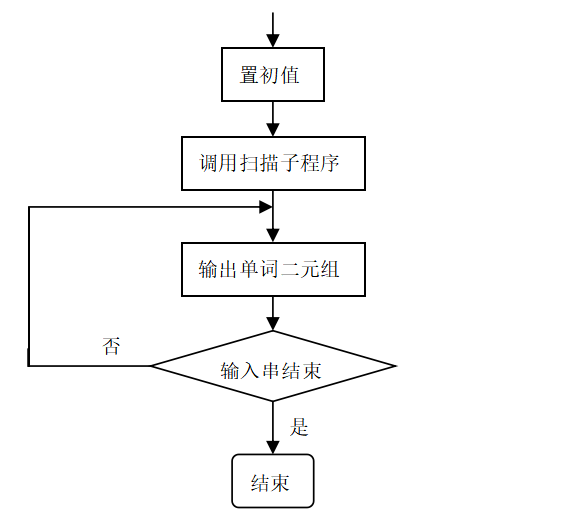
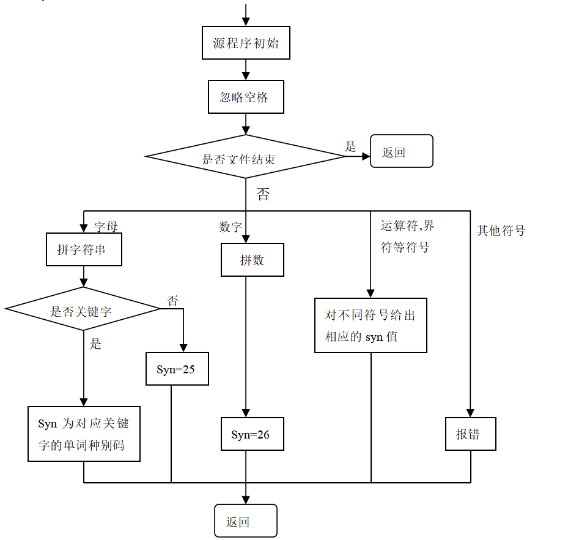
** **

**图 16NFA类图 图 17DFA类图**

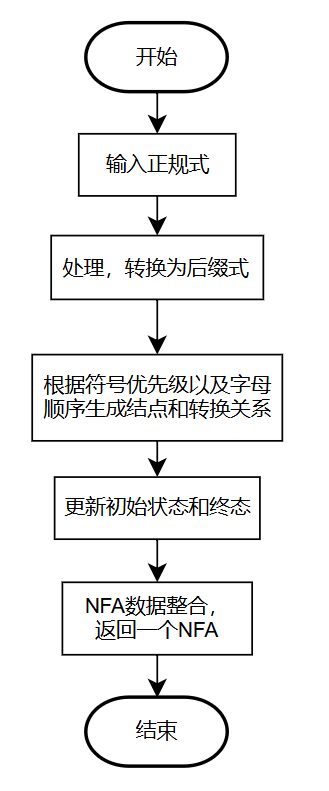
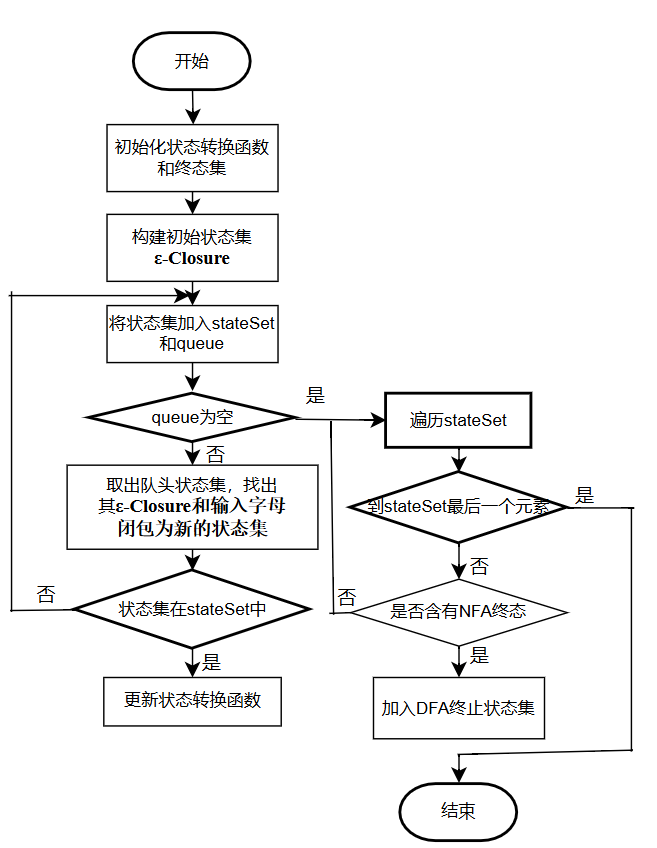
****

**图 18交互图**

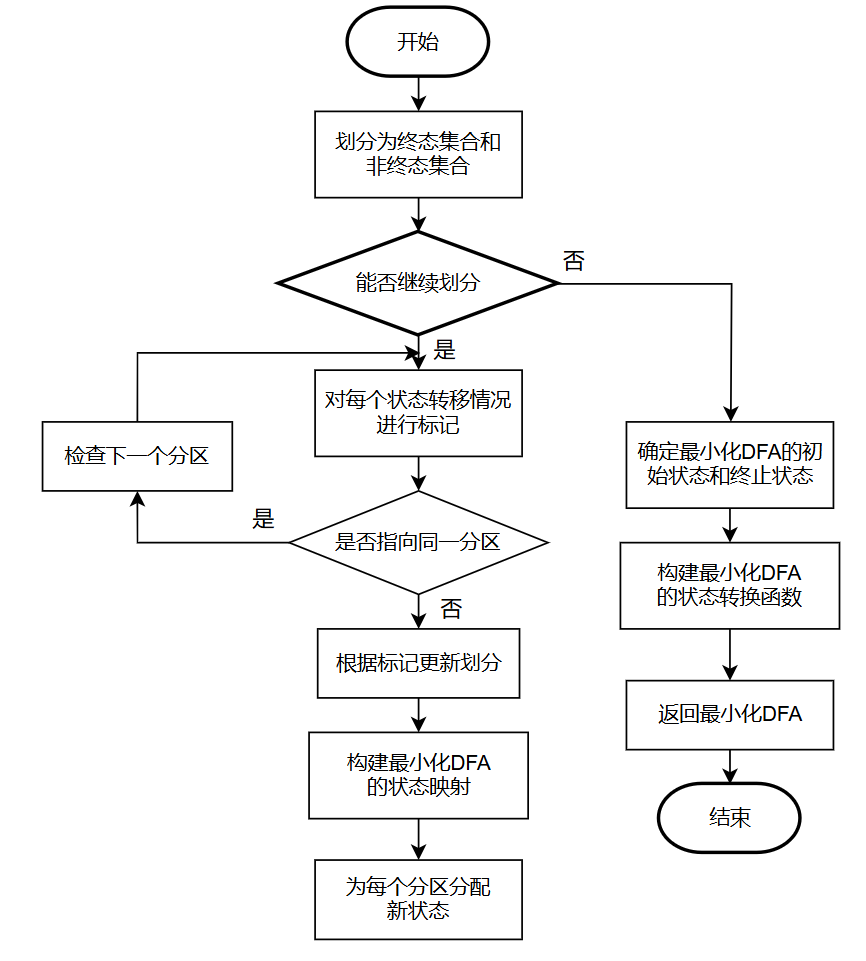
1. **主要算法描述**

**图 19词法分析器主函数流程图 图 20扫描子函数getToken()流程图**

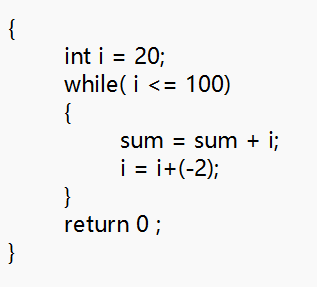
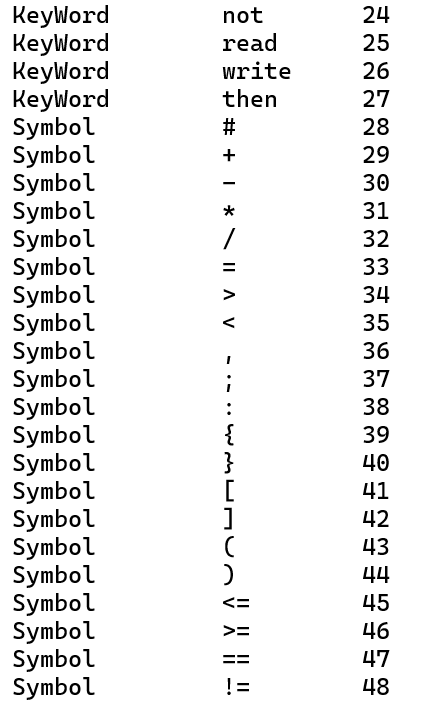
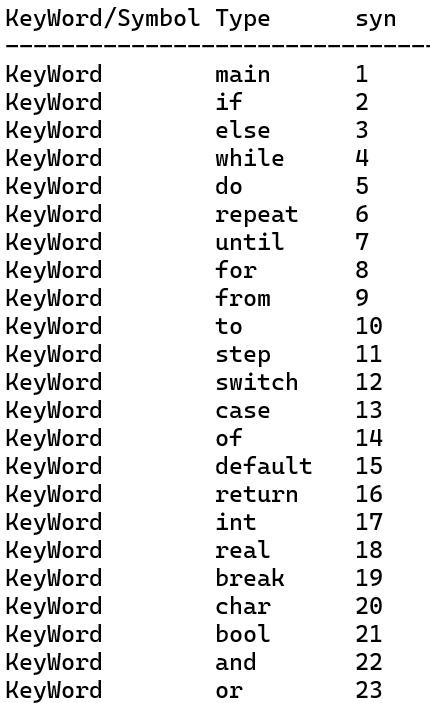
** **

**图 21正规式转NFA流程图 图 22 NFA确定化为DFA流程图**

****

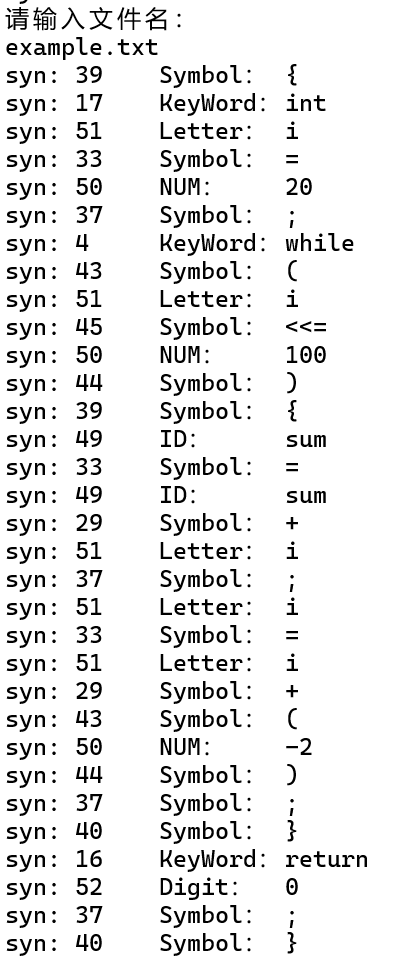
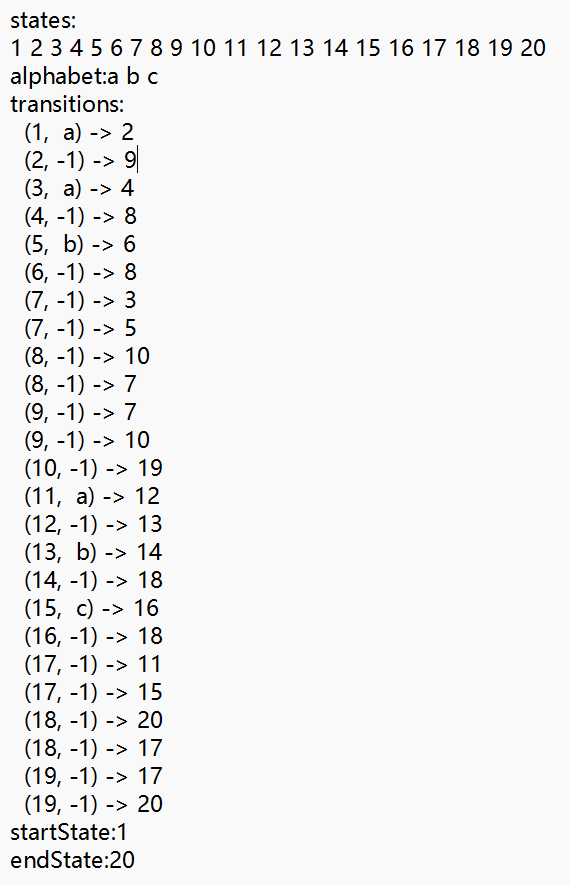
**图 23 DFA最小化流程图**

1. **测试数据与测试效果**

****

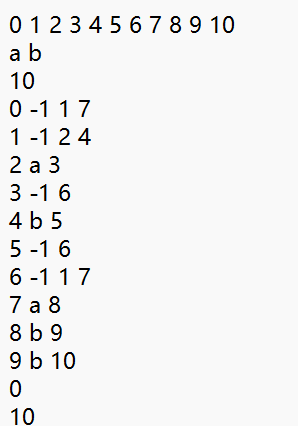
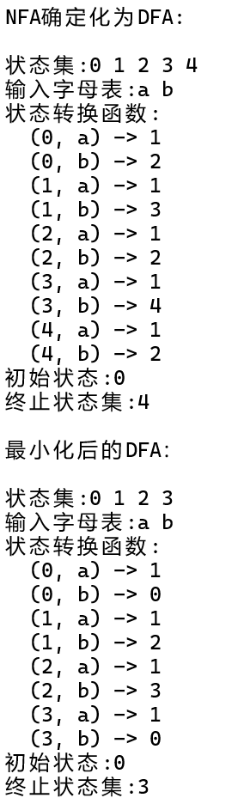
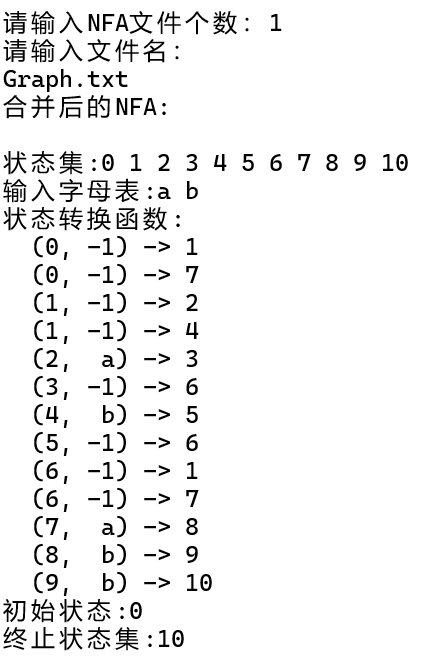
**图 24 example.txt 图 25输出符号表**

**2.** **RE转NFA测试内容**：a(a|b)\*((a|b)|c)\*

** **

**图 26源程序的输出对象 图 27 RE转NFA.结果**

**3.NFA确定化为DFA并最小化测试内容与结果**

**** 

**图 28 Graph.txt(NFA描述文件) 图 29 NFA转DFA并实现最小化结果**

1. **实验总结**

**1.遇到的问题及解决的方法**

**（1）问题：设计词法分析器时，希望它能够识别负的整型变量。但是出来的结果错误，存在重复输出现象。**

**解决方法：多次尝试发现指针指向错误，而且遇到负号时应该先跳过去计算常量。**

**（2）问题：在将两个NFA合并，形成新的NFA这一步中，合并结果总是错误，不能输出正确的状态与转移函数。**

**解决方法：通过创建一个新的状态集合来存储合并后的状态，确定每次生成的新起始状态编号为两个NFA状态集合中最大值加1。同时注意初始状态和接受状态的合并处理，确保新的NFA具有正确的初始和接受状态定义。**

**（3）问题：RE转换成NFA时不明白怎样生成新的状态和如何更新转移函数。**

**解决方法：查询网上资料，结合课件，理解要先将RE转换成后缀表达式。便于计算机理解。再通过算法将其转换为NFA。**

**（4）问题：输出NFA、DFA的转移函数时‘-1’不也能正确输出。**

**解决方法：这个问题一直存在，应该先对输入字符进行判断，是‘-1’单独输出。**

**（5）问题：DFA最小化算法的实现存在困难，对于状态等价性的判断和接受状态转移到新的分区难以完成。**

**解决方法：查询网上关于DFA最小化的资料，进一步理解实现过程。首先根据接受状态和非接受状态将DFA的状态划分为两个初始集合。接着反复检查每个集合中的状态对于每个输入符号的转移是否都落在相同的分区中，如果不同，则进一步划分集合。通过这种迭代的方式，最终实现DFA的最小化。**

**2.实验后的收获**

**（1）对词法分析原理和有穷自动机的理论知识的理解进一步加深。将NFA确定化为DFA是为了消除NFA的不确定性，使得自动机的行为更加明确和易于分析，DFA的最小化则是为了在不改变自动机识别语言的前提下，简化自动机的结构，提高效率。**

**（2）在完成实验的过程中锻炼了我的编程能力。熟悉如何用复杂的数据结构来表示不同的对象，以及如何设计高效的算法来实现操作。**

**（3）在面对实验中出现的各种问题时，通过不断地分析问题、查阅资料和尝试不同的解决方法，培养了独立解决问题的能力。**

**（4）在建立分析模型和设计模型的过程中，体会到了软件设计的过程，进一步掌握了各种图的表示方法，这种思维方式在今后的学习和研究中具有你重要意义。**

**3.实验建议**

**在讲解理论知识时，可以用代码示例来展示如何将理论知识转化为实际的代码，加深同学们的理解。对于一些关键的代码片段，可以详细讲解其实现思路和技巧，帮助我们更好掌握编程实现的方法。**