**武汉大学计算机学院**

**本科生课程设计报告**

**基于WPF的CMM语言解释器系统总体设计与实现**

专 业 名 称 ：软件工程卓越工程师班

课 程 名 称 ：解释器构造实践

指 导 教 师 ：刘纪平 讲师

学 生 学 号 ：2017302580305

学 生 姓 名 ：祝捷

二○一九年十一月

**郑 重 声 明**

本人呈交的设计报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本设计报告不包含他人享有著作权的内容。对本设计报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本设计报告的知识产权归属于培养单位。

本人签名： 祝捷 日期： 2019年11月10号

摘要

XXXX实验的实验目的是XXXX。

实验设计主要遵循XXXX。

实验内容主要包括：

实验结论为XXXX

……

……

……

**关键词：**关键词1；关键词2；关键词3

（黑体小4） （宋体小4）

**目录**

**1.实验目的和意义**

1.1 实验目的 ………………………………………………………………………1

1.2 运行环境………………………………………………1

**2.实验设计**

2.1系统架构概述与前端设计………………………………………………………… 35

2.2词法分析板块………………………………………………………………………… 37

2.3实验方案………………………………………………………………………… 39

**3实验总结**

2.1概述……………………………………………………………………………… 35

2.2实验原理………………………………………………………………………… 37

2.3实验方案………………………………………………………………………… 39

**结论** …………………………………………………………………………………… 57

**参考文献 ………………………………………………………………………………** 59

**附录 ……………**………………………………………………………………………… 72

**一、实验目的和意义**

**1.1实验目的**

本实验的主要目的是设计一个CMM语言的解释器，分为四个部分进行完成，实验一部分是设计CMM语言解释器的总体架构，设计并编制调试一个分析单词的词法分析器，从而加深对词法分析原理的理解。

**1.2运行环境**

本次实验构造的解释器系统主要运行在基于Microsoft公司开发的WPF框架的PC端桌面应用软件上，开发工具是Visual Studio，主要语言是C#和XAML，操作系统是Windows10，实验一主要完成了项目交互页面设计与词法分析板块。

**二、实验设计**

**2.1系统架构概述与前端设计**

**2.1.1 系统总体框架综述**

本次实验的解释器系统主要采用了基于微软的WPF框架并以C#为主要语言进行开发，使用了GitHub上的一个WPF前端组件库Material Design In XAML Toolkit。然后我的解释器系统主要可以分为：词法分析，语法分析，语义分析，三个步骤板块进行依次执行，还提供了包括导出语法树，导出中间结果等可选功能。设计上美观大方，页面提供了可供编辑的代码编辑区易于调整，使得用户可以享受从打开，编辑，调试，运行，保存等一整套连贯式的程序编写流程。

项目系统源代码主要包含了页面前端代码，业务逻辑代码，以及工具异常类等辅助代码。具体来说，页面前端代码主要是MainWindows.xaml，主要是基于WPF的一个页面，采取了Material Design in XAML提供的一些Button field toolbar等组件进行美化，并与后台代码进行绑定，执行不同的任务。具体请参考后文中前端设计与框架部分的介绍。

而后台业务逻辑简单地说就是对于不同的点击事件进行不同的处理，具体代码实现出于代码可读性考虑分别放在对应的分析板块的代码中，另外自定义了一个LexException类用于报错信息整合处理，使得调试更加方便，也使得代码逻辑一目了然。

代码真正执行处理任务的代码大部分都整合在Token.cs Tools.cs等工具类和业务板块类中，比如Token类的getNextToken就执行了对于一个string对象分析出下一个token的这个词法分析任务，所以比较长，具体逻辑可以结合我定义的词法状态机进行理解。

**2.1.2 系统前端设计与框架**

本次实验的解释器系统前端主要采用了基于微软的WPF框架以C#为主要语言的开发模式，使用了GitHub上的一个WPF前端组件库Material Design In XAML Toolkit，这个组件库是基于Google提出的Material Design设计思想开发的。。我选择这个框架主要是因为它提供的组件风格统一，美观大方，为WPF桌面端应用开发提供了便利，也能进一步优化我们的界面效果，改善用户交互体验。

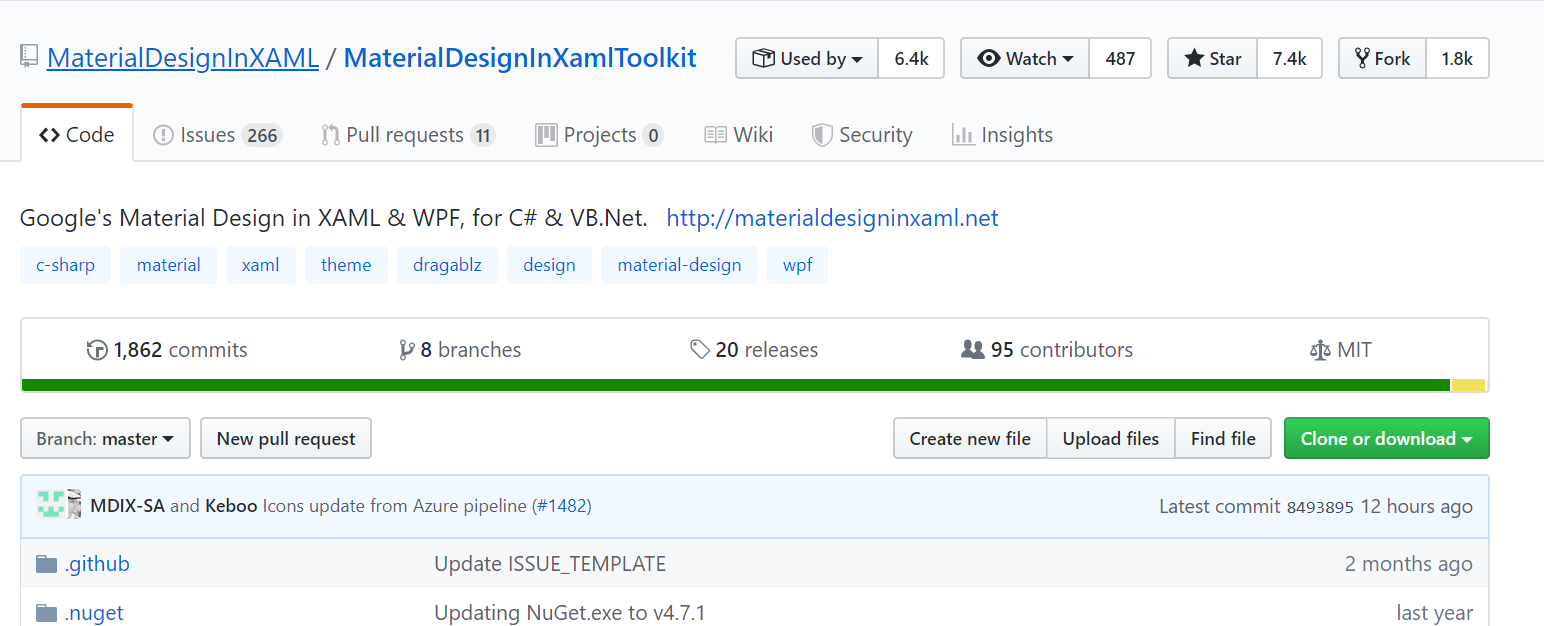


图1 Material Design In XAML Toolkit开源组件库

具体开发是在Visual Studio中基于WPF进行设计调整，WPF跟我们熟悉的Winform一样提供了非常便利的所看即所得的开发效果，可以很方便的对于组件的页面效果，绑定事件等等进行调整。

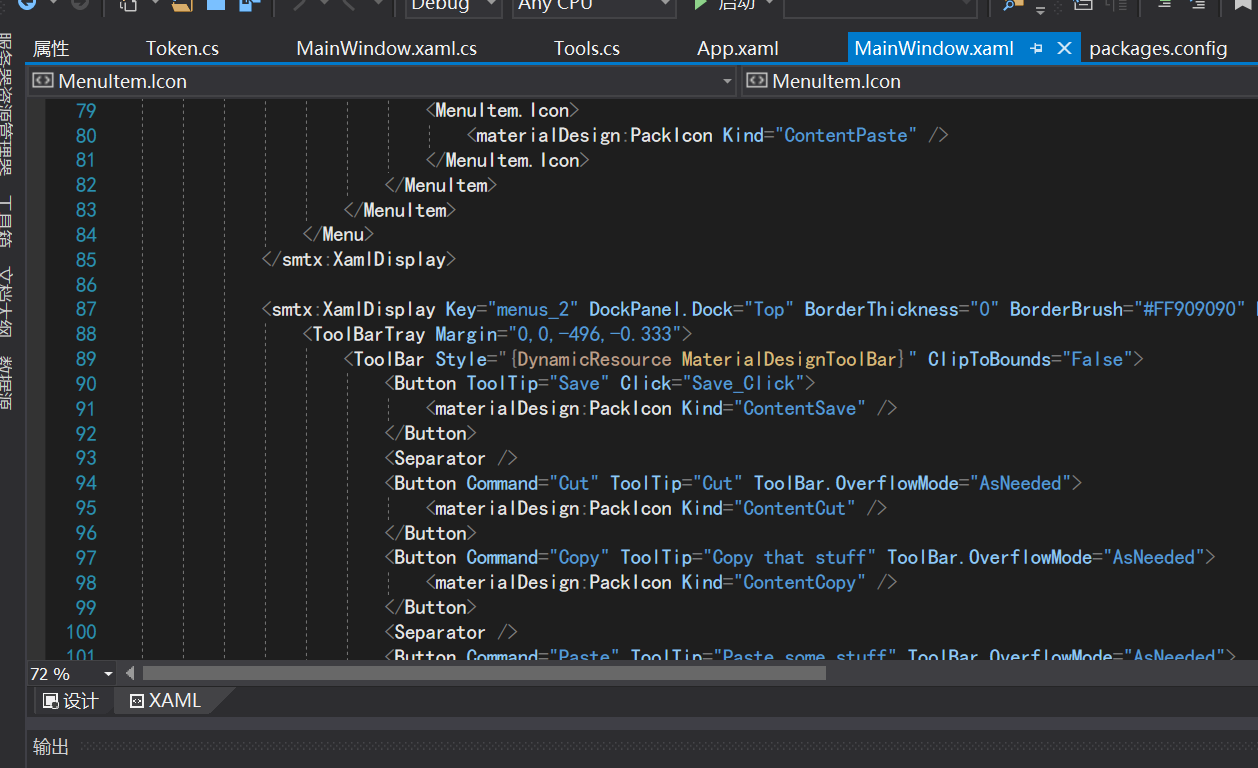


图2 WPF提供的基于xaml页面设计

**2.1.3 系统前端开发的问题与解决**

1. 缺少相关教程和文档说明

通过查看demo中的效果与源码进行比对，结合WPF的一些组件使用方法，去尝试理解各种组件的使用方式，然后多看看demo如何使用的，合适的时候可以避免采取一些相对简单的方式也能实现需求。

2. 组件操作复杂，代码写起来很繁琐复杂

同时，对于一些非常复杂的组件，在直接操作组件的一些方法和属性不方便的时候，采取工具类的思想，封装一些访问和修改的方法，提升了代码可读性，避免每次都是一大段代码进行访问，使方法过于复杂。

3. IO输入输出时出现了乱码

通过设定utf-8编码格式保证编码不会出现问题

**2.2词法分析板块**

**2.2.1 语言形式化描述**

下面采用文法的方式对定义的CMM语言进行描述：

关于文法的简要说明：，即四元组形式，表示非终结符号集，表示终结符号集，P是表示产生式的有穷非空集合，S为开始符号，至少出现在一条产生式的左部。

**Production产生式:**

**2.2.2 单词编码表（码表）**

采取一符一码的形式，根据下面的码表产生对应的属性字token并赋予类别码。

|  |  |
| --- | --- |
| 单词符号 | 类别码 |
| if | 1 |
| else | 2 |
| int | 3 |
| while | 4 |
| read | 5 |
| write | 6 |
| real | 7 |
| true | 8 |
| false | 9 |
| null | 10 |
| + | 11 |
| - | 12 |
| \* | 13 |
| / | 14 |
| = | 15 |
| += | 16 |
| -= | 17 |
| \*= | 18 |
| /= | 19 |
| < | 21 |
| > | 22 |
| == | 23 |
| <> | 24 |
| >= | 25 |
| <= | 26 |
| != | 27 |
| ! | 28 |
| && | 29 |
| || | 30 |
| ( | 31 |
| ) | 32 |
| { | 33 |
| } | 34 |
| [ | 35 |
| ] | 36 |
| , | 47 |
| ; | 48 |
| 常数 | 49 |
| 标识符 | 50 |

**2.2.3 词法分析程序的数据结构与算法流程说明**

下面先展示系统的完整架构uml类图：

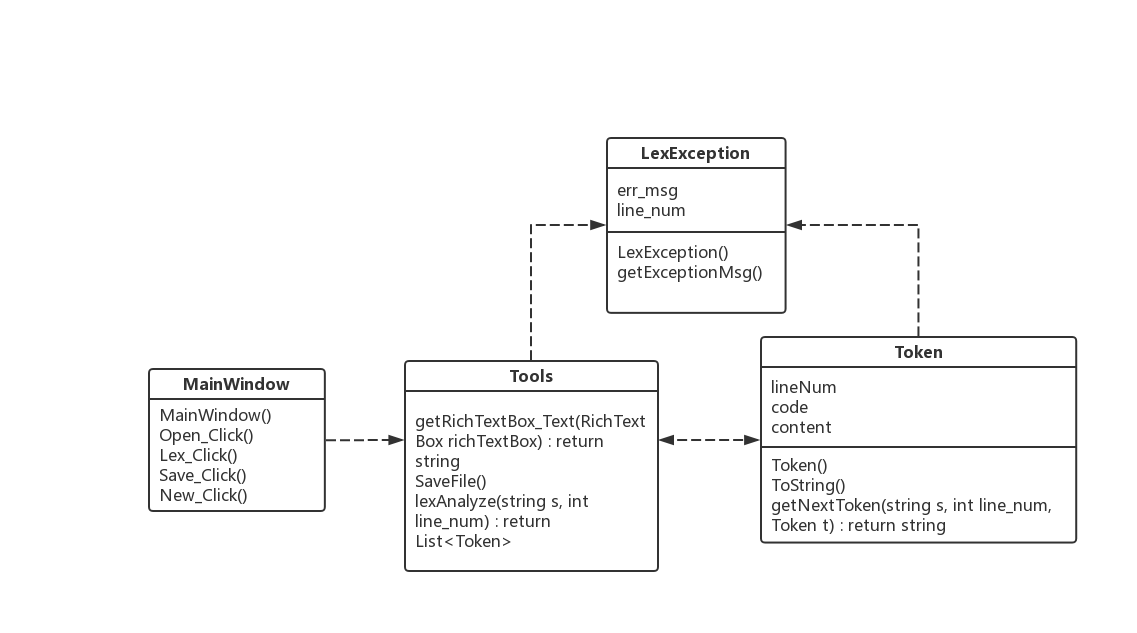


图2.2.3.1 系统uml类图

下面是词法分析的状态机图：

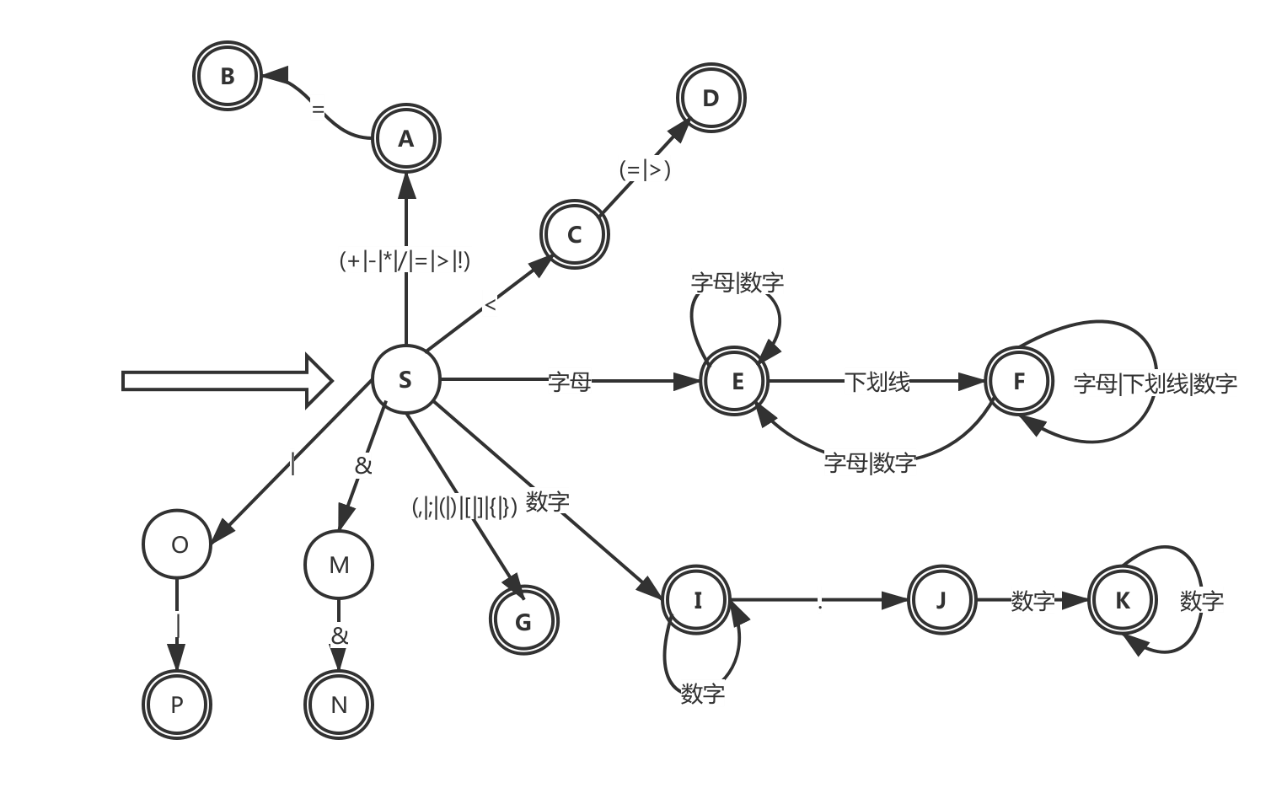


图2.2.3.2 状态机图

下面是完整的词法分析程序流程图：(出于开发效率和便于调整的考虑，后期对于词法程序测试修改的内容没有反映到这个程序流程框图中，也就是说，下面的这些流程图并不一定完全对应程序，具体请参考源代码，uml系统类图2.2.3.1或上面的状态机图2.2.3.2，这两个图保证是最新的)

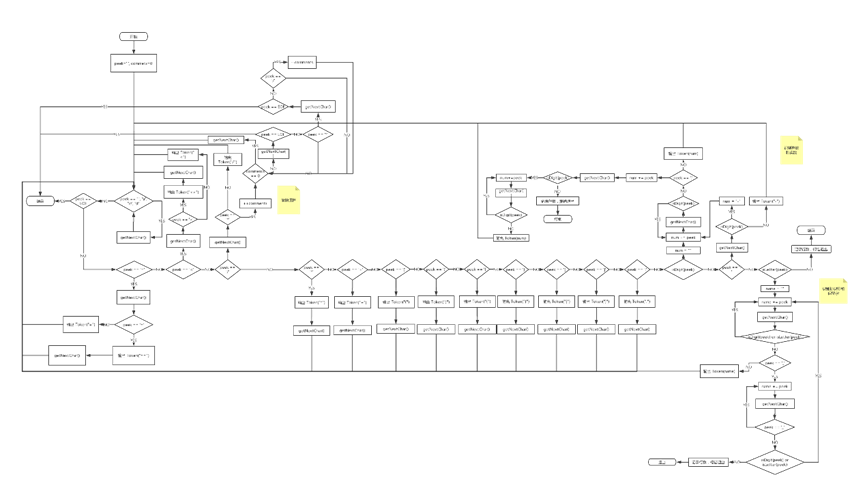


图2.2.3.3 完整流程图（老图）

对于整数/实数的识别流程：

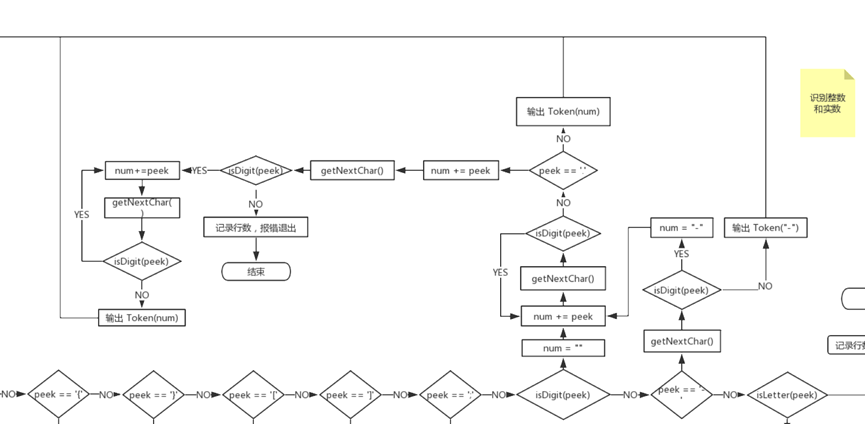


图2.2.3.4 整数实数识别流程图（老图）

对于标识符的识别流程：

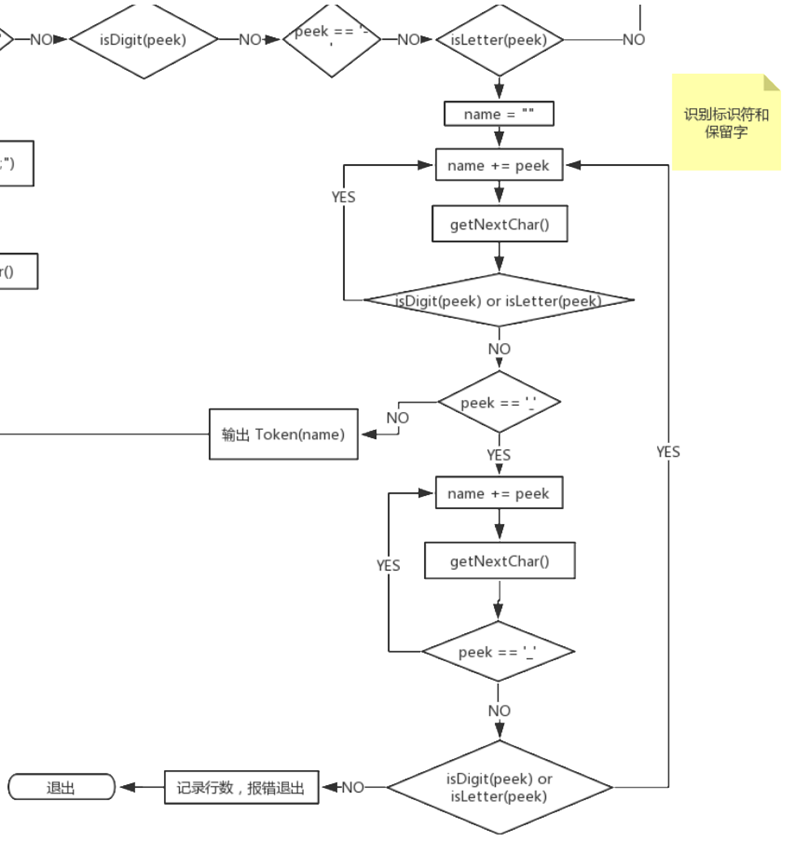


图2.2.3.5 标识符识别流程图（老图）

下面我将会结合系统uml架构图来具体介绍程序中使用到的数据结构与算法流程：

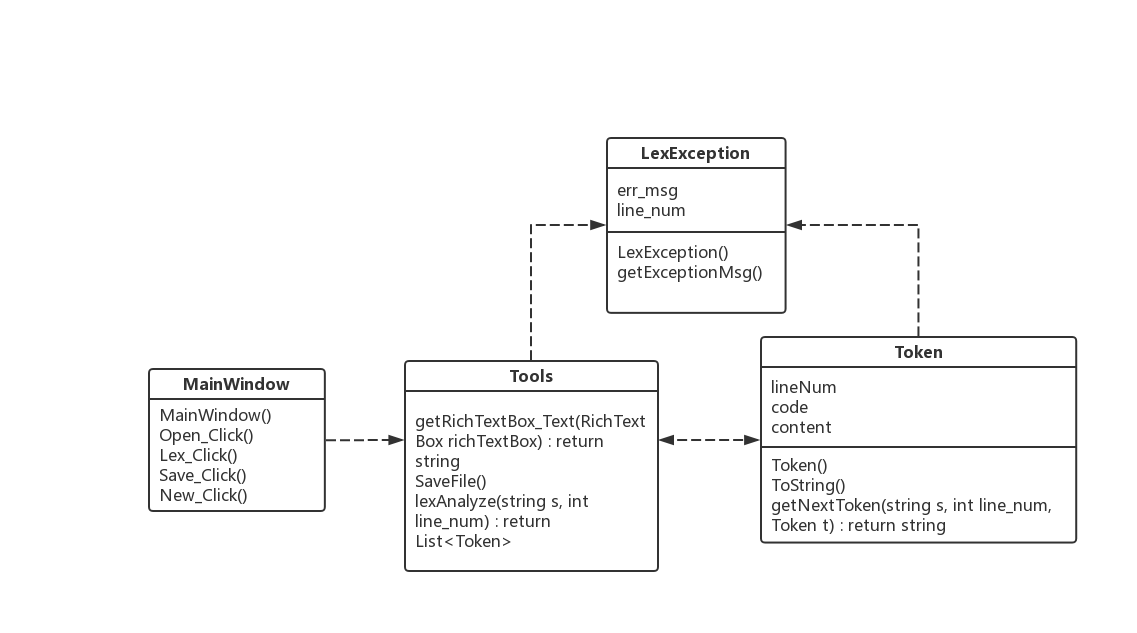


图2.2.3.6 系统uml类图

首先词法分析中使用到的主要数据结构是一个List<List<Token>>嵌套的自定义类型的有序线性表tokens\_Lst，词法分析的基本流程就是MainWindow收到一个点击事件，然后调用Tools类的lexAnalysis方法，该方法会读入textBox中已经导入的文件代码，并逐行进行分析。

具体逐行分析的过程是通过调用Token类的getNextToken方法来实现的，该方法接受一个string类型（该行代码），进行分析，若分析出一个token就返回该token，并把token加到List中去，当该行代码分析完则把List添加到tokens\_list的嵌套List中，读入下一行。若词法分析过程出错，getNextToken会抛出一个自定义的LexException异常，该异常中存有错误信息和错误发生的函数，上层调用它的函数lexAnalysis捕捉到异常后，会立刻终止该行的词法分析，提示错误信息，并继续其他行的词法分析。

最后，我们把完整的tokens\_List中存放的token属性字，按行号打印出来显示在结果区域。

**2.2.4 词法分析程序的使用，测试与运行效果**

启动词法分析程序，在工具栏点击open按钮，从本机打开任意一个CMM的文件，程序会自动读入文件，并导入到codebox源程序编辑区，可以对源程序做修改并保存。

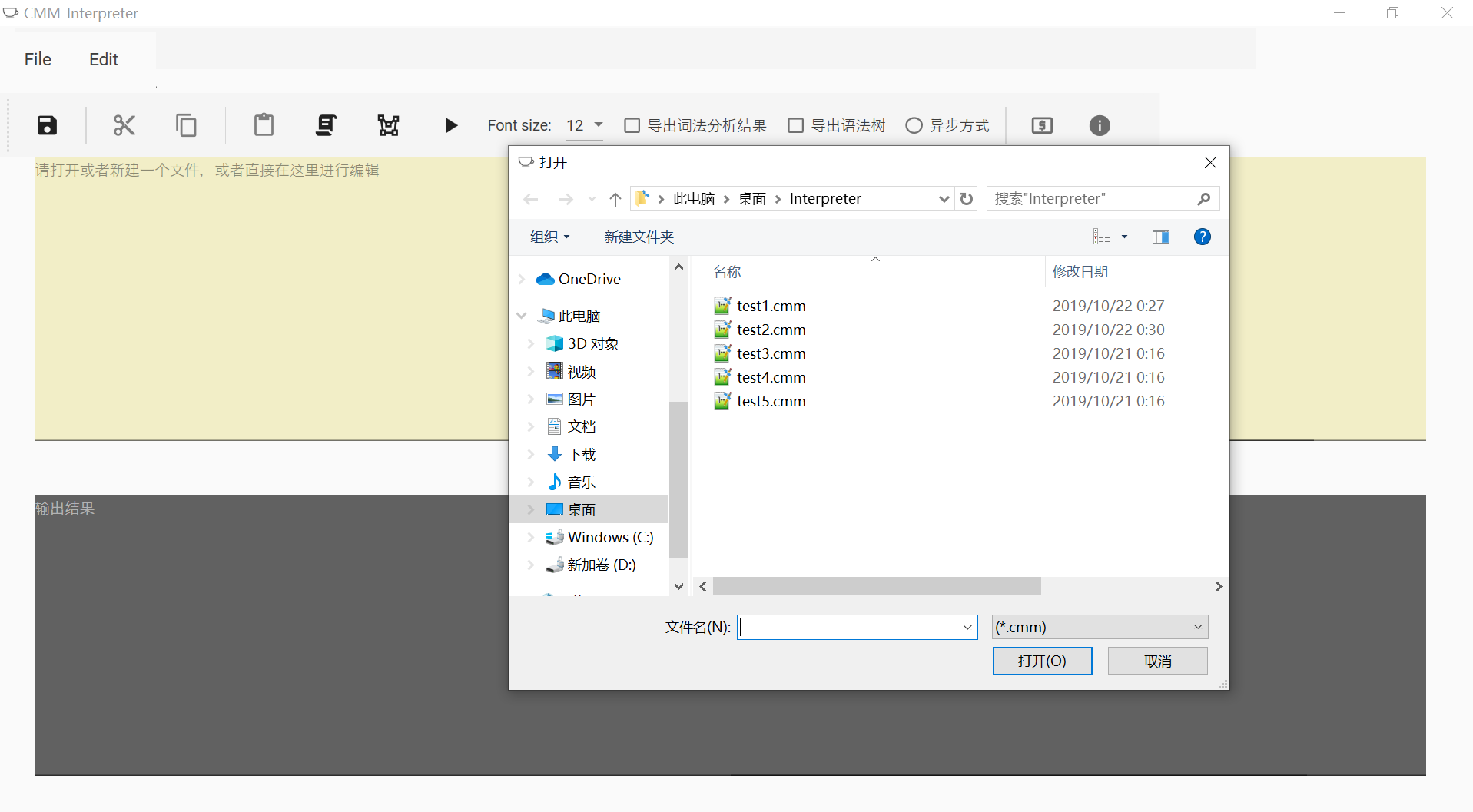


图2.2.4.1 打开文件

点击词法分析按钮可进行词法分析，程序会读入编辑区已导入的源程序代码，进行词法分析。分析结果显示在下方的分析结果区，如果出现错误，错误信息会以MessageBox的形式弹窗提示，同时出现错误的代码行会被跳过，我们的程序会继续分析下一行代码，最终同样会把结果显示到结果区域。

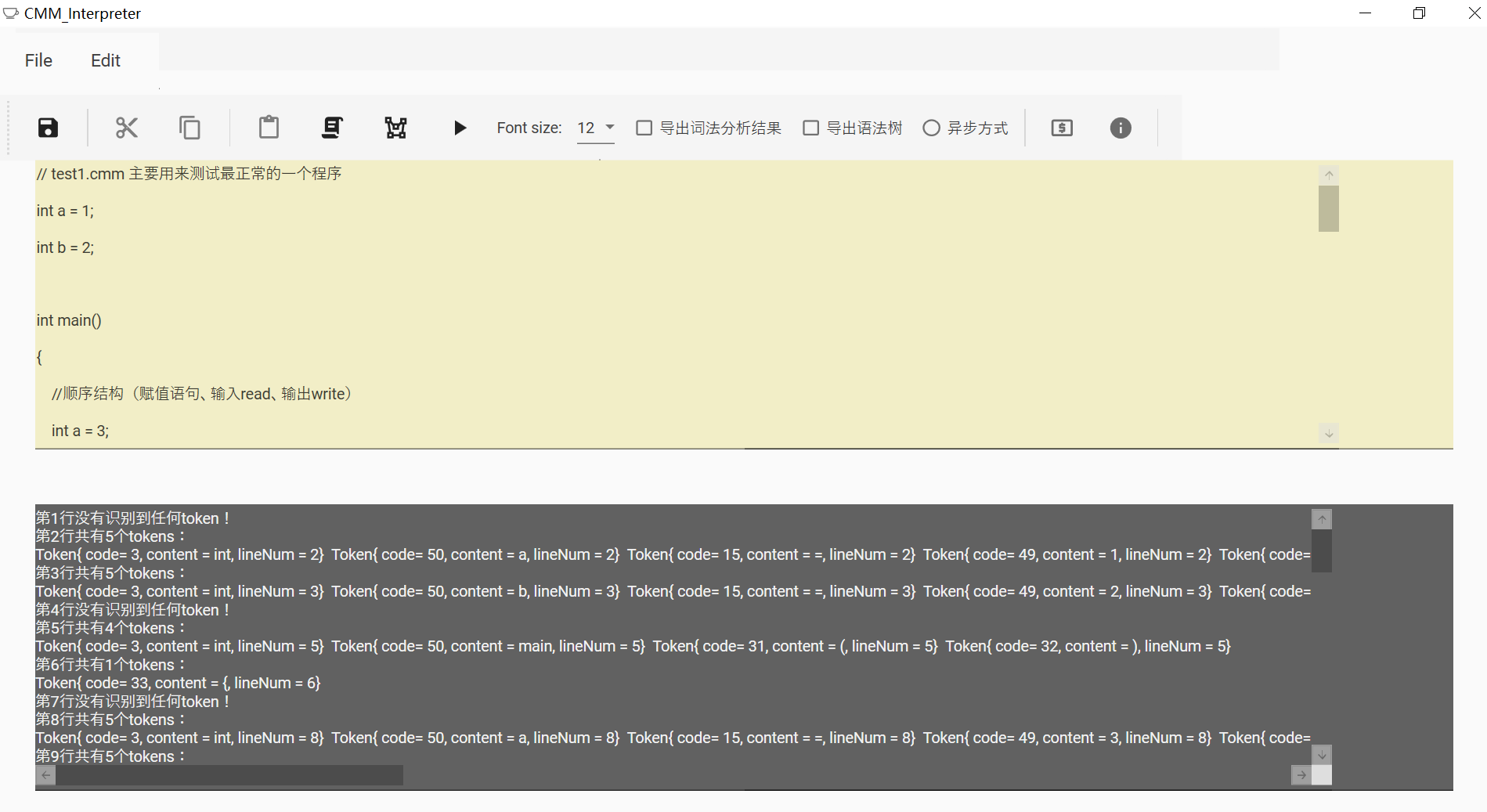


图2.2.4.2分析成功显示结果

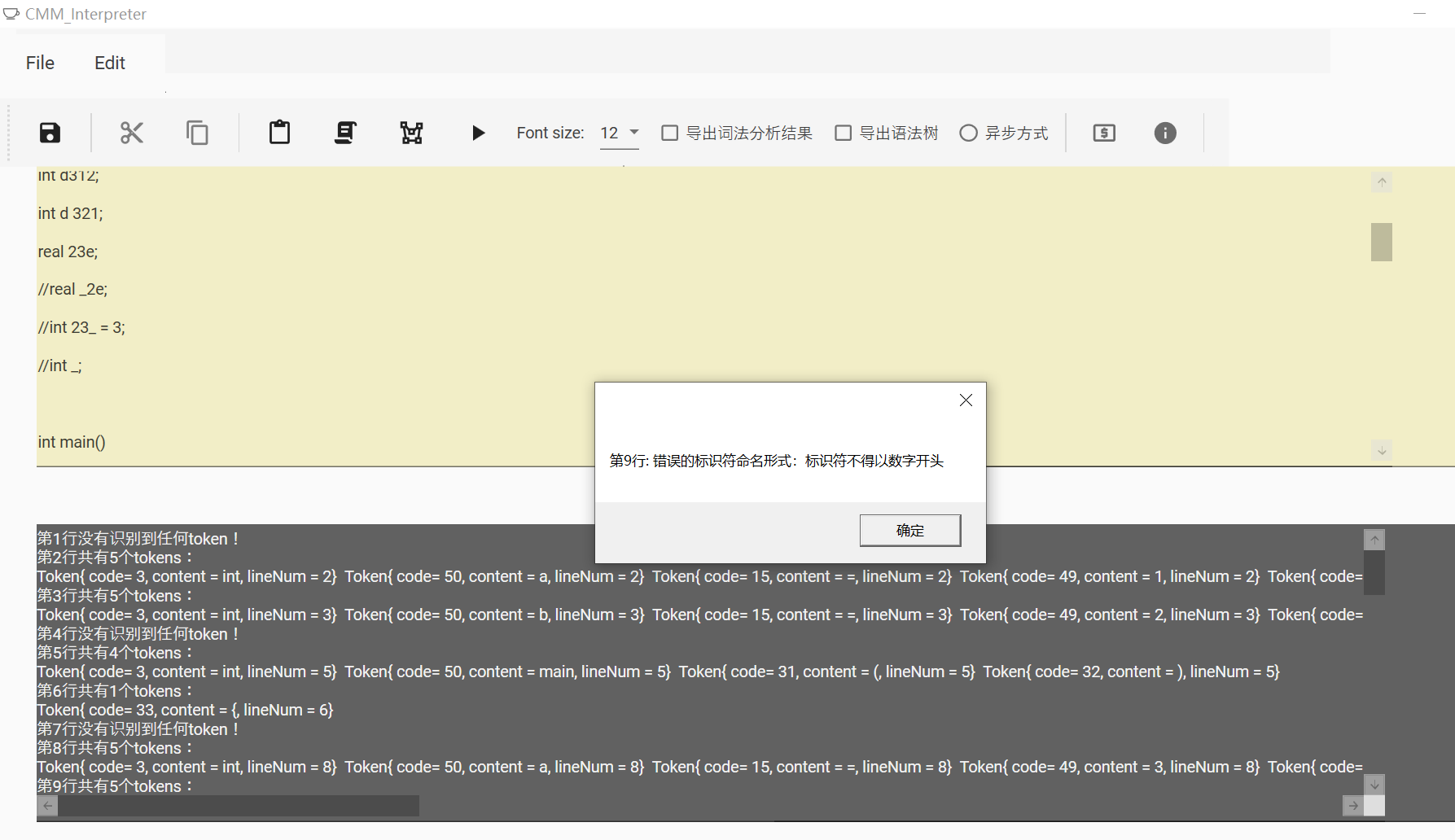


图2.2.4.3显示报错信息

下面介绍我对词法分析程序进行的有关测试：

测试原则：遵循二八原则，也就是把80%的时间或者资源用在20%的重点模块上，重点测试模块中20%的重要模块。来达到测试效率和资源配置的最佳的比例。我主要采取单元测试的方法。部分测试样例如下：

第一个测试：对正常程序的完整测试

**2.2.5 问题、困难与解决**

问题1：需求复杂，不断迭代：词法分析程序需要不断迭代，可能会漏掉一些识别的符号，或者可能以前设计的状态机之后会进行修改，可能测试中发现的问题需要及时调整，而且测试要方便且易于显示错误信息。

解决：

基于面向对象思想设计，结构清晰。控制好数据层，页面层，业务逻辑层，实现与处理层等等层次之间的调用关系，便于之后调整代码。设计自定义异常类显示报错，同时也便于软件测试。

总体上说，运用这些设计思想都极大增强了代码的可读性，鲁棒性，符合高内聚低耦合的设计模式观点。

问题2：词法设计中，减号和负号如何处理？

解决：

实际上这个问题我也是第一次遇见，当时编译原理课程的编译器我没有处理负数的情况，但是这次解释器势必要解决这个问题。

开始我觉得不是很复杂，觉得-减号后面有数字就是负号，就识别成负数，否则就是减号，但很明显是错误的（比如a-2）。之后我又尝试了几种方法来识别，发现测试时均不能很好的达到目标，属于语法的范畴。

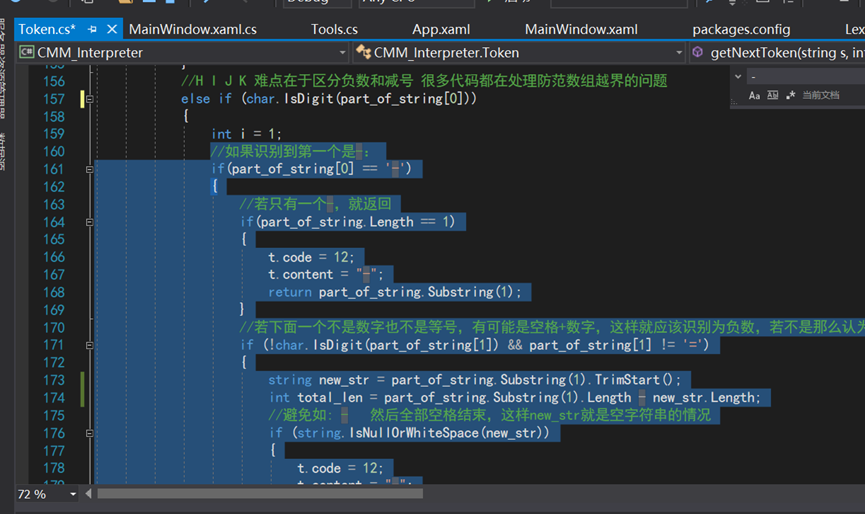
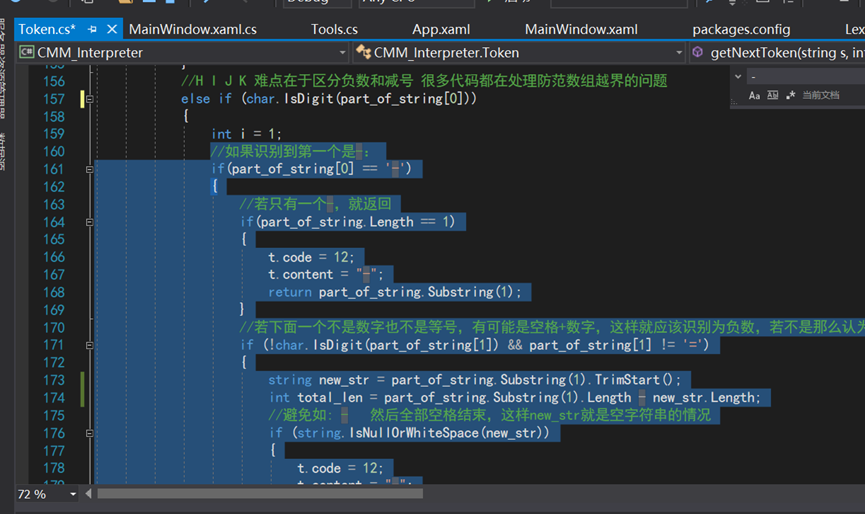


图2.2.5.1 识别负数部分的代码（后来删去了）

后来我在C++环境进行尝试，发现是可以接受任意多个-减号的，所以我最后就决定把减号负号都识别成一种符号，具体情形到语法分析时在进行讨论。

问题3：数组越界异常

参考文献示例：

**参考文献** (黑体小2)

[1] 戴军，袁惠新.膜技术在含油废水处理中的应用[J].膜科学与技术，2002，22（2）：59-64.

[2] 毛侠，孙云.和谐图案的自动生成研究[A].第一届中国情感计算及智能交互学术会议论文集[C].北京：中国科学院自动化研究所，2003：277-279.

[3] 王湛.膜分离技术基础[M].北京:化学工业出版社，2000：14-21，30.

[4] 张志祥. 间断动力系统的随机扰动及其在守恒律方程中的应用[D].北京:北京大学数学学院,1998.

[5] World Health Organization. Factors regulating the immune response: report of WHO Scientific Group[R]. Geneva: WHO, 1970.

[6] 河北绿洲生态环境科技有限公司.一种荒漠化地区生态植被综合培育种植方法:中国,01129210.5[P].2001-10-24.

[7] GB/T16159-1996,汉语拼音证词法基本规则[S].北京：中国标准出版社，1996.

[8] 毛侠.情感工学破解“舒服之谜”[N].光明日报，2004-04-17（B1）.

[9] 陈剑.上博简《民之父母》“而得既塞於四海矣”句解释[EB/OL］.简帛研究网站，http://www.bamboosilk.org/Wssf/2003/chenjian03.htm．2003-01-18.

( 宋体小4)

……

……

……