使用实例

说明：以下使用实例均截自于课程设计项目报告

一、词法分析器使用实例

例 1：简单的复合语句测试（这里以 while 为例）

输入：

```
int main(){
int a=0;
for(int i=0;i<10;i++){
a=a+1;
}
}
```

输出：

```
(line 1,type,int)
(line 1,identifier,main)
(line 1,delimiters,()
(line 1,delimiters,))
(line 1,delimiters,{)
(line 2,type,int)
(line 2,identifier,a)
(line 2,binary_operation,=)
(line 2,constant,0)
(line 2,delimiters,;)
(line 3,keyword,for)
(line 3,delimiters,()
(line 3,type,int)
(line 3,identifier,i)
(line 3,binary_operation,=)
(line 3,constant,0)
(line 3,delimiters,;)
(line 3,identifier,i)
(line 3,binary_operation,<)
(line 3,constant,10)
(line 3,delimiters,;)
(line 3,identifier,i)
(line 3,unary_operation,++)
(line 3,delimiters,))
(line 3,delimiters,{)
(line 4,identifier,a)
(line 4,binary_operation,=)
(line 4,identifier,a)
(line 4,binary_operation,+)
(line 4,constant,1)
(line 4,delimiters,;)
(line 5,delimiters,})
(line 6,delimiters,})
```

第一组测试说明，该词法分析器能正常的处理简单复合语句，并给出相应的三元式 token 序列。

例 2：常量及正负号处理

输入：

```
int main(){
int a=-3;
int b=2E+10;
a=$10+12i;
}
```

输出：

```
(line 1,type,int)
(line 1,identifier,main)
(line 1,delimiters,()
(line 1,delimiters,))
(line 1,delimiters,{)
(line 2,type,int)
(line 2,identifier,a)
(line 2,binary_operation,=)
(line 2,constant,-3)
(line 2,delimiters,;)
(line 3,type,int)
(line 3,identifier,b)
(line 3,binary_operation,=)
(line 3,constant,2E+10)
(line 3,delimiters,;)
(line 4,identifier,a)
(line 4,binary_operation,=)
(line 4,constant,$10+12i)
(line 4,delimiters,;)
(line 5,delimiters,})
```

第二组测试说明，该词法分析器能够正常的处理各种常量，包括负数常量、科学计数法以及复数。这里需要强调的是：为了和 error 类型进行区分，这里约定复数常量必须以$开头来作为复数常量的标识。

例 3：词法错误识别

输入：

```
int main(){
int 3asd=1;
}
```

输出：

```
(line 1,type,int)
(line 1,identifier,main)
(line 1,delimiters,()
(line 1,delimiters,))
(line 1,delimiters,{)
(line 2,type,int)
(line 2,error,3asd)
发现词法错误！数字不能作为标识符的首字母
```

第三组测试说明，该词法分析器能够识别出词法单词错误，并给出错误所在地方和错误原因。

二、语法分析器使用实例

例 1：for 循环语句识别

输入：

```
int main(){
int a=1;
for(int i=0;i<10;i++){
a=a+1;
}
}
```

输出：

```
YES
```

第一组测试说明，该语法分析器能够识一般的 for 循环语句。

例 2：while 循环语句识别

输入：

```
int main(){
int a=1;
while(a<10){
a++;
}
}
```

输出：

```
YES
```

第二组测试说明，该语法分析器能够识一般的 while 循环语句。

例 3：if 语句识别

输入：

```
int main(){
int a=1;
if(a<10){
int b=1;
}
}
```

输出：

```
YES
```

第三组测试说明，该语法分析器能够识一般的 if 语句。

例 4：嵌套语句识别（以 while-if 嵌套为例）

输入：

```
int main(){
int a=1;
while(a<10){
if(a==2){break;}
a=a+1;
}
}
```

输出：

```
YES
```

第四组测试说明，该语法分析器能够识嵌套语句。

例 5：语法报错（这里以缺少分隔符为例）

输入：

```
int main(){
int a=1;
int b=2
a=a+1;
}
```

输出：

```
NO
----line information----
line 3: int b = 2
line 4: a = a + 1 ;
line 5: }
------------------------
error line: 4
error value: a
----expected symbol-----
;
------------------------
```

第五组测试说明，该语法分析器能够识语法错误，并给出错误行数、错误位置以及大概错误原因。

三、语义分析器使用实例

例 1：if 语句的分析

输入：

```
int a=1;
if(a<10){a=a+1;}
int b=2;
```

输出：

```
100:a=1
101:if not a<10 goto 104
102:temp0=a+1
103:a=temp0
104:b=2
105:
```

第一组测试说明，该语法分析器能够识一般的 if 语句，并输出其四元式代码。

例 2：while 语句的分析

输入：

```
int a=1;
while(a<10){a=a+1;}
int b=2;
```

输出：

```
100:a=1
101:if not a<10 goto 105
102:temp0=a+1
103:a=temp0
104:goto 101
105:b=2
106:
```

第二组测试说明，该语法分析器能够识一般的 while 语句，并输出其四元式代码。

例 3：for 语句的分析

输入：

```
int a=1;
int b=2;
for(int i=1;i<10;i++){a=a+1;}
```

输出：

```
100:a=1
101:b=2
102:i=1
103:if not i<10 goto 109
104:temp0=a+1
105:a=temp0
106:temp1=i+1
107:i=temp1
108:goto 103
109:
```

第三组测试说明，该语法分析器能够识一般的 for 语句，并输出其四元式代码。

四、QT 图形界面展示

此处的功能测试不再给出具体例子，而是展示具体的图形化界面并介绍每个分析器的具体使用方法和流程。

4.1 界面展示

1. 主界面

2. 词法分析器界面

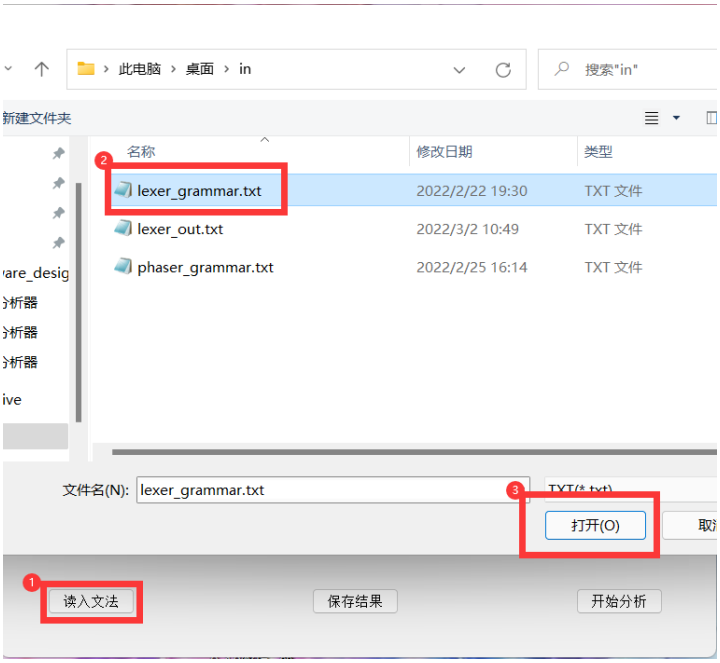
3. 语法分析器界面

4. 语义分析器界面

4.2 词法分析器功能测试

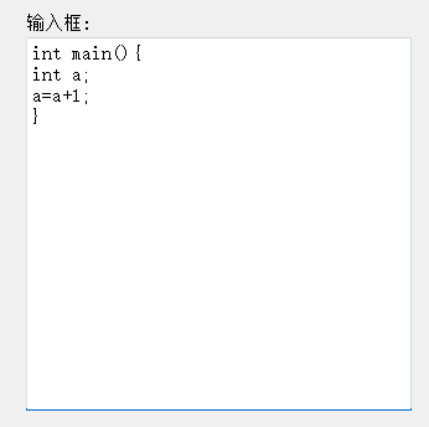
Step1：文法读入

如下图，点击“读入文法”，然后再文件对话框中选择相应文件并点击“打开”即可。



Step2：源代码的输入

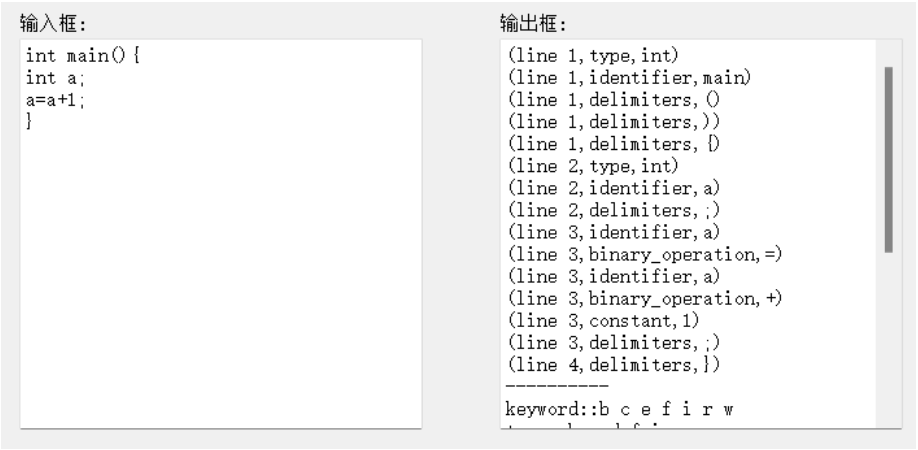
如下图，在输入框中输入想要分析的源代码即可。



Step3：开始分析

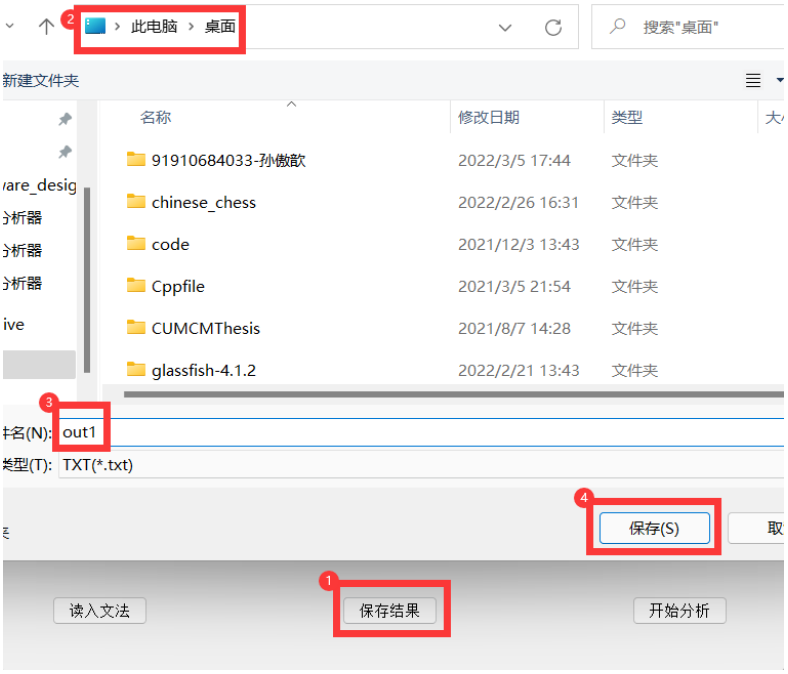
如下图，点击右下角的“开始分析”按钮即可开始进行词法分析。

如下图，分析结果将写入到输出框中。



Step4：结果保存

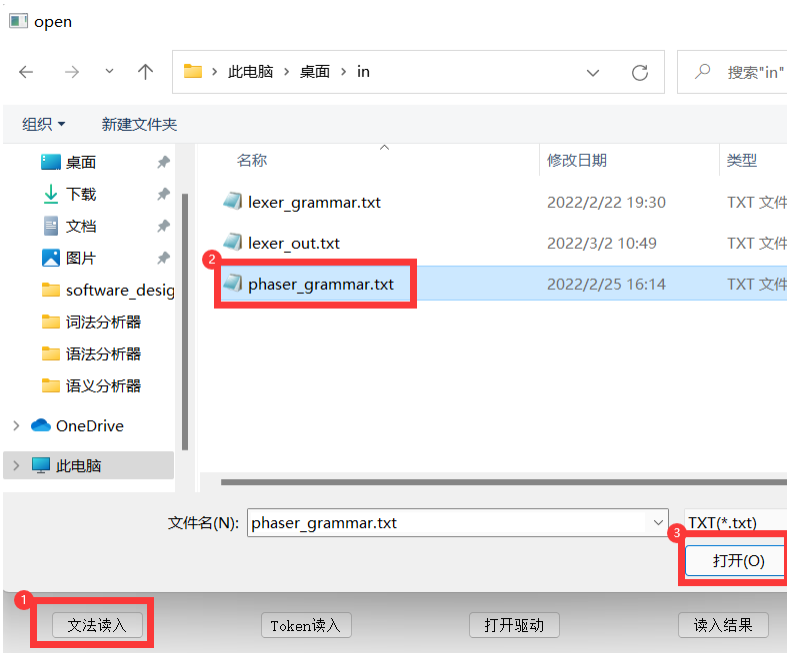
如下图，点击“保存结果”按钮，选择保存路径，输入文件名，点击保存即可。



4.3 语法分析器功能测试

Step1：文法读入

如下图，点击“文法读入”，然后在文件对话框中选择相应文件并点击“打开”即可。



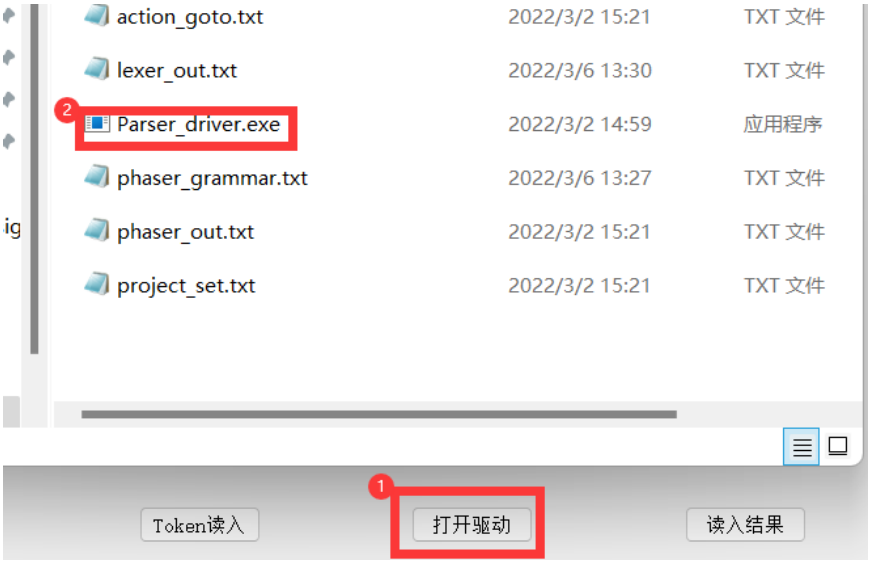
Step2：Token 读入

和**文法读入步骤**完全一致，在此便不再赘述。

Step3：打开驱动程序，进行分析

如下图，点击“打开驱动”按钮，然后双击“Parser_driver.exe”文件即可

开始分析。

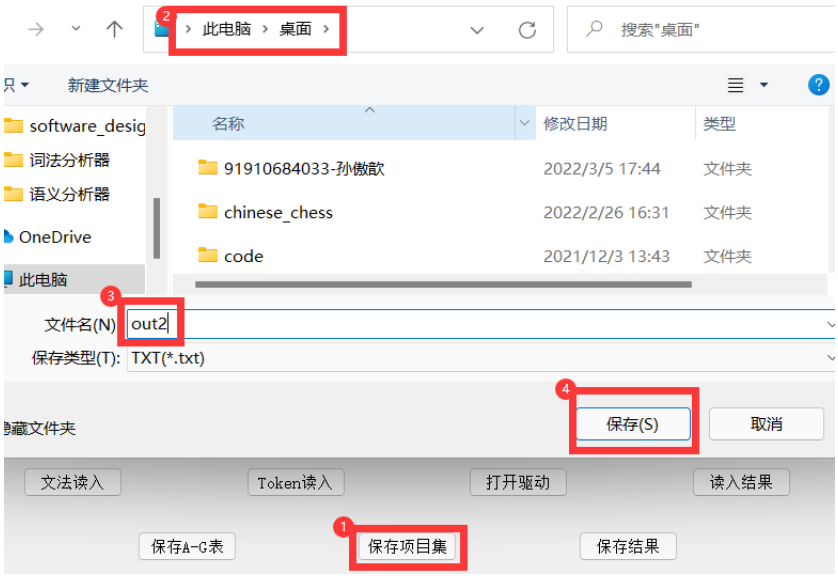


Step4:读入结果

如下图，分析完成后，点击“读入结果”按钮，程序运行结果将显示于三个输出框内。

Step5：结果保存

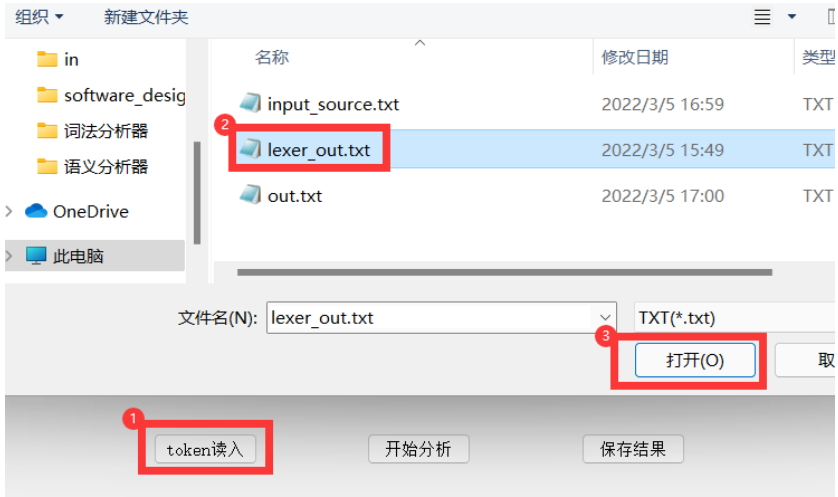
如下图，点击任意一个结果保存按钮（这里以保存项目集为例），选择保存路径，输入文件名，点击保存即可。



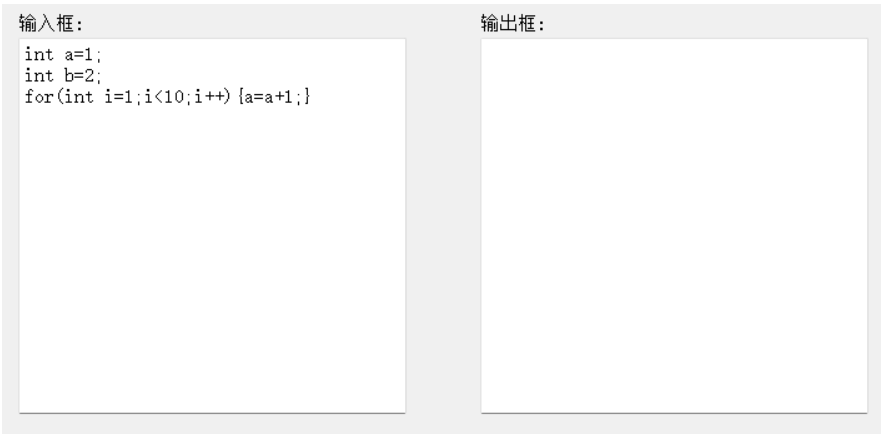
4.4 语义分析器功能测试

Step1：token 读入及翻译

如下图，点击“token 读入”，然后在文件对话框中选择相应文件并点击“打开”即可。



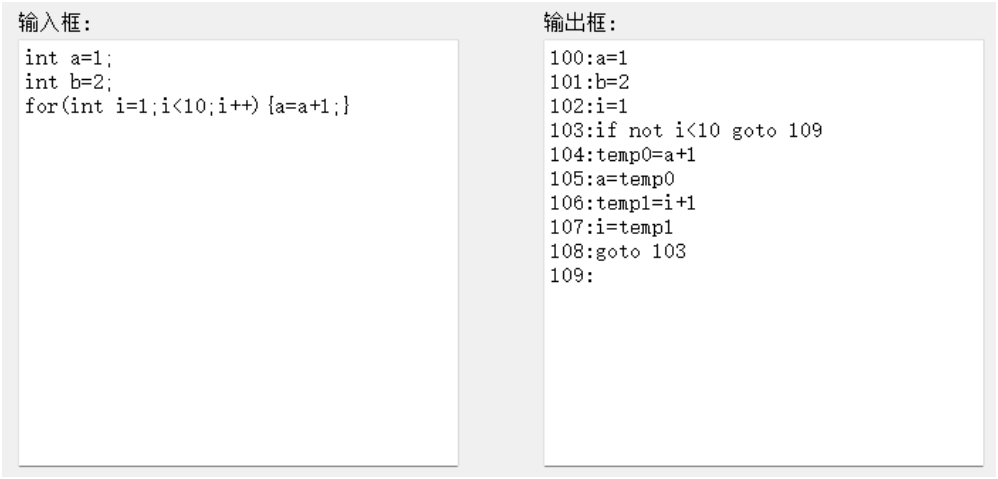
如下图，读入 token 后，会将翻译后的源代码写入至输入框。当然也可以不读如 token 文件，直接在输入框进行编辑。



Step2：开始分析

如下图，点击右下角的“开始分析”按钮即可开始进行词法分析。

如下图，分析结果将写入到输出框中。



Step3：结果保存

如下图，点击“保存结果”按钮，选择保存路径，输入文件名，点击保存即可。