

中国矿业大学计算机学院

系统软件开发实践报告

课程名称：系统软件开发实践

实验名称：实验一 利用 Flex_Bison 构造编译器

学生姓名：陈柏翰

学生学号：02140385

专业班级：计算机科学与技术 2014-4 班

任课教师：张博老师

实验一 利用 Flex/Bison 构造编译器

一 实验目的

- a) 利用 Flex 设计一个扫描器程序，用于计算一个文件中的字符数，单词数和行数。

二 分析 flex 代码

```
%{                               \\\  第一段：是 C 和 Lex 的全局声明 int nchar,
int nchar, nword, nline; \\\  nchar 字符数 nword 单词数 nline 行数

%}

%%                               \\\  段与段之间以%%来分界
                               \\\  第二段：包括模式（C 代码）

\n { nline++; nchar++; } \\\  若是回车 则行数+1 字符数+1
[^\t\n]+ { nword++; nchar += yyleng; } \\\  若不是回车不是空格 单词数
                                     +1。字符数=字符数+ yyleng 给出匹配模式的长度。
. { nchar++;                     \\\  匹配除了 n 的任意字符 字符数+1
%%                               \\\  段与段之间以%%来分界
                               \\\  第三段：是补充的 C 函数

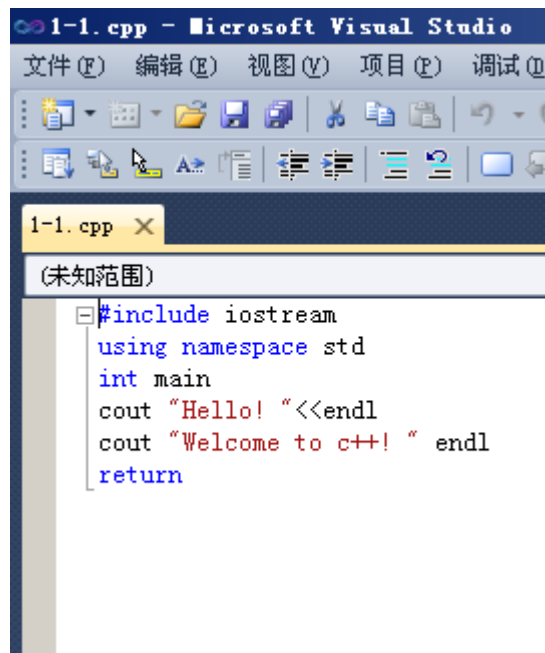
void main()
{
    yylex();                     \\\  yylex ( ) 函数开始分析 由 lex 自动生成
    printf("%d\t%d\t%d\n", nchar, nword, nline); \\\  打印结果
}
int yywrap()                     \\\  在下文解释
{
    return 1;
}
```

\\ yywrap() 这一函数在文件（或输入）的末尾调用。如果函数的返回值是 1，就停止解析。因此它可以用来解析多个文件。代码可以写在第三段，这就

能够解析多个文件。方法是使用 yyin 文件指针（见上表）指向不同的文件，直到所有的文件都被解析。最后，yywrap() 可以返回 1 来表示解析的结束。

三 分析程序输出结果

程序输出结果为 103 个字符 17 个单词 5 行



```
1-1.cpp - Microsoft Visual Studio
文件(F) 编辑(E) 视图(V) 项目(P) 调试(U)
1-1.cpp
(未知范围)
#include iostream
using namespace std
int main
cout "Hello! " << endl
cout "Welcome to c++! " endl
return
```

第一个单词为 #include	单词数+1=1	字符数+8=8
接着是空格	字符数+1=9	
下一个单词为 iostream	单词数+1=2	字符数+8=17
接着是 \n	字符数+1=18	行数+1=1
下一个单词为 using	单词数+1=3	字符数+5=23
接着是空格	字符数+1=24	
下一个单词为 namespace	单词数+1=4	字符数+9=33
接着是空格	字符数+1=34	
下一个单词为 std	单词数+1=5	字符数+3=36
接着是 \n	字符数+1=37	行数+1=2

下一个单词为 int	单词数+1=6	字符数+3=40
接着是空格	字符数+1=41	
下一个单词为 main	单词数+1=7	字符数+4=44
接着是 \n	字符数+1=45	行数+1=3
下一个单词为 cout	单词数+1=8	字符数+4=49
接着是空格	字符数+1=50	
下一个单词为 "Hello !	单词数+1=9	字符数+7=57
接着是空格	字符数+1=58	
下一个单词为 "<<endl	单词数+1=10	字符数+7=65
接着是 \n	字符数+1=66	行数+1=4
下一个单词为 cout	单词数+1=11	字符数+3=69
接着是空格	字符数+1=70	
下一个单词为 "Welcome	单词数+1=12	字符数+8=78
接着是空格	字符数+1=79	
下一个单词为 to	单词数+1=13	字符数+2=81
接着是空格	字符数+1=82	
下一个单词为 c++ !	单词数+1=14	字符数+4=86
接着是空格	字符数+1=87	
下一个单词为 "	单词数+1=15	字符数+1=88
接着是空格	字符数+1=89	
下一个单词为 endl	单词数+1=16	字符数+4=93
接着是 \n	字符数+1=94	行数+1=5
下一个单词为 return	单词数+1=17	字符数+6=103

三 在 WINDOWS 环境下的实验步骤

- b) Windows 环境下安装 Flex。（一直选择 next，安装完成出现 GnuWin32 文件）





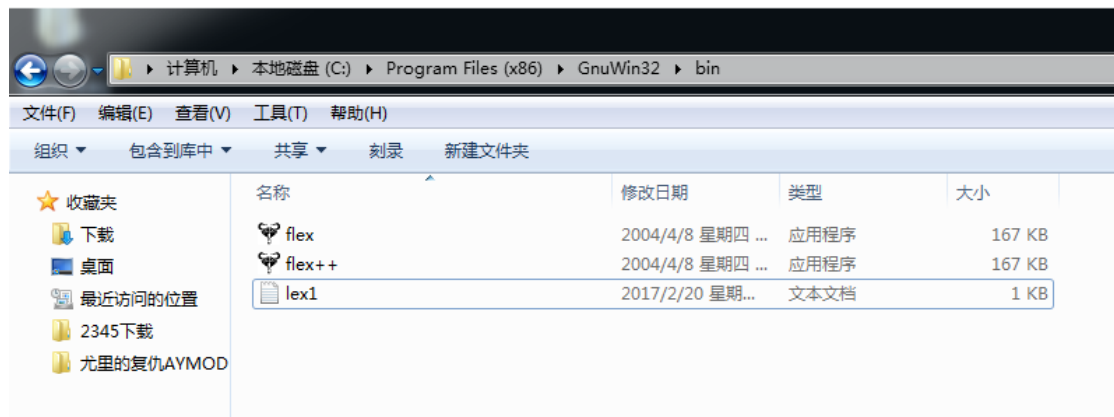
c) Windows 环境下安装 NOTEPAD++。



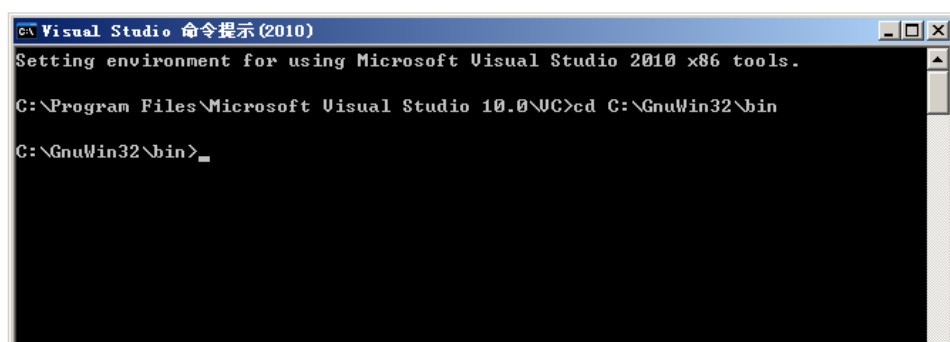
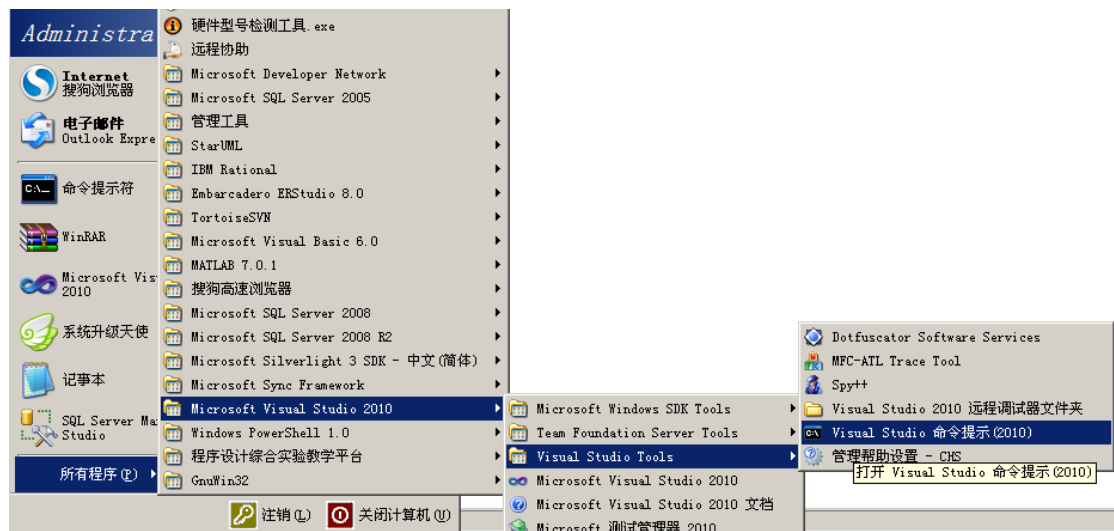


- c) 将源程序（代码如下）保存为 lex1.l 文件放在 GnuWin32 文件夹下的 bin 目录下，如下：

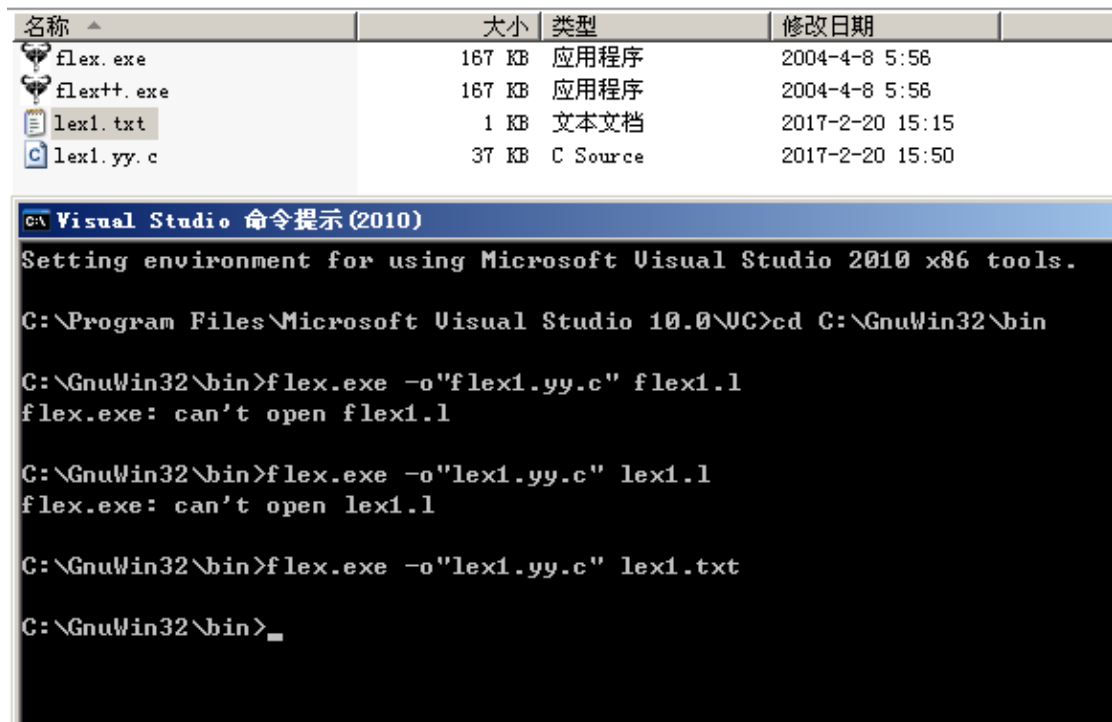
```
*C:\Program Files (x86)\Notepad++\change.log - Notepad++ [Administrator]
文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P)
change.log x
1  %{
2  int nchar, nword, nline;
3  %}
4  %%
5  \n { nline++; nchar++; }
6  [^ \t\n]+ { nword++, nchar += yyleng; }
7  . { nchar++; }
8  %%
9  void main()
10 {
11 yylex();
12 printf("%d\t%d\t%d\n", nchar, nword, nline);
13 }
14 int yywrap()
15 {
16 return 1;
17 }
```



d) 打开 Visual Studio 2008 命令行，进入 flex 安装目录，输入
> cd C:\GnuWin32\bin



E) 调用 flex.exe , 输入 > flex.exe -o"lex1.yy.c" lex1.l,生成 lex1.yy.c 文件



F) 接着在命令窗口输入 `cl lex1.yy.c`,得到如下图结果

名称	大小	类型	修改日期
flex.exe	167 KB	应用程序	2004-4-8 5:56
flex++.exe	167 KB	应用程序	2004-4-8 5:56
lex1.txt	1 KB	文本文档	2017-2-20 15:15
lex1.yy.c	37 KB	C Source	2017-2-20 15:50
lex1.yy.exe	54 KB	应用程序	2017-2-20 15:52
lex1.yy.obj	11 KB	Object File	2017-2-20 15:52


```
C:\ Visual Studio 命令提示 (2010)
Setting environment for using Microsoft Visual Studio 2010 x86 tools.

C:\Program Files\Microsoft Visual Studio 10.0\VC>cd C:\GnuWin32\bin

C:\GnuWin32\bin>flex.exe -o"flex1.yy.c" flex1.l
flex.exe: can't open flex1.l

C:\GnuWin32\bin>flex.exe -o"lex1.yy.c" lex1.l
flex.exe: can't open lex1.l

C:\GnuWin32\bin>flex.exe -o"lex1.yy.c" lex1.txt

C:\GnuWin32\bin>cl lex1.yy.c
用于 80x86 的 Microsoft (R) 32 位 C/C++ 优化编译器 16.00.30319.01 版
版权所有 (C) Microsoft Corporation。保留所有权利。

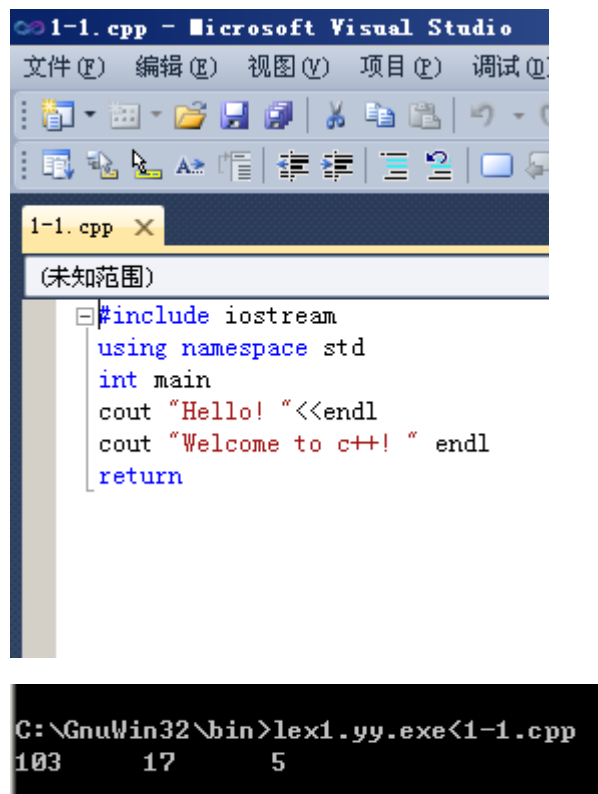
lex1.yy.c
Microsoft (R) Incremental Linker Version 10.00.30319.01
Copyright (C) Microsoft Corporation. All rights reserved.

/out:lex1.yy.exe
lex1.yy.obj

C:\GnuWin32\bin>
```

此时目标目录中生成了 `lex1.yy.exe` 和 `lex1.yy.obj` 两个文件

G) 调用 lex1.yy.exe,对 1-1.cpp (代码如下) 进行词法分析 , 输出如下结果



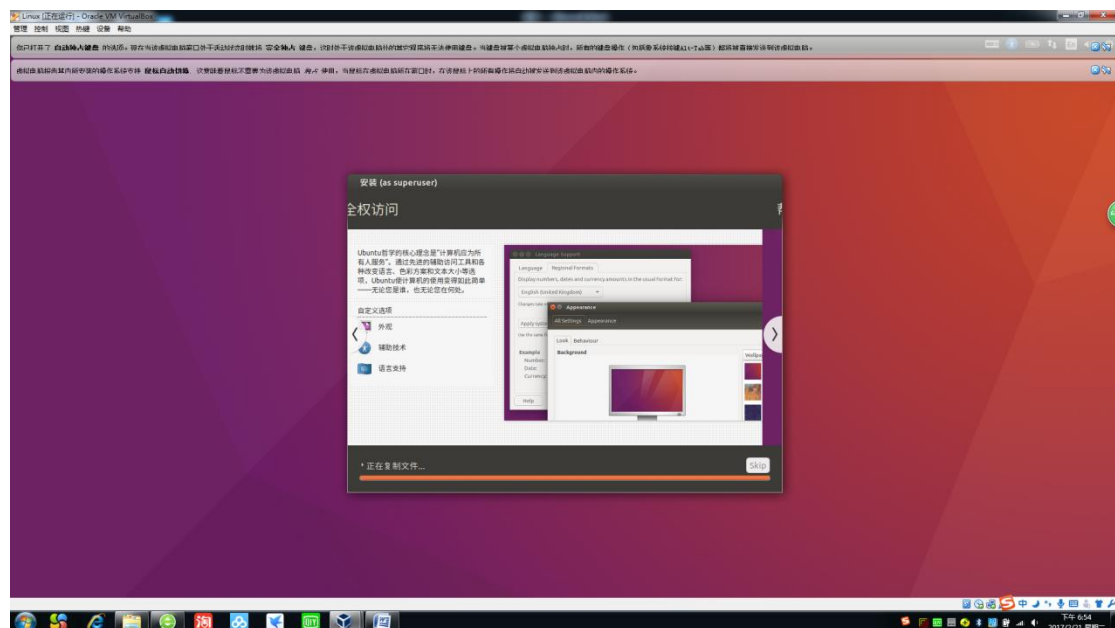
The image shows a screenshot of Microsoft Visual Studio. The top part displays the code for 1-1.cpp, which includes iostream, uses namespace std, and has a main function that prints "Hello!" and "Welcome to c++!". The bottom part shows a terminal window with the command C:\GnuWin32\bin>lex1.yy.exe<1-1.cpp and its output: 103 17 5.

```
#include iostream
using namespace std
int main
cout <<"Hello!" <<endl
cout <<"Welcome to c++!" <<endl
return
```

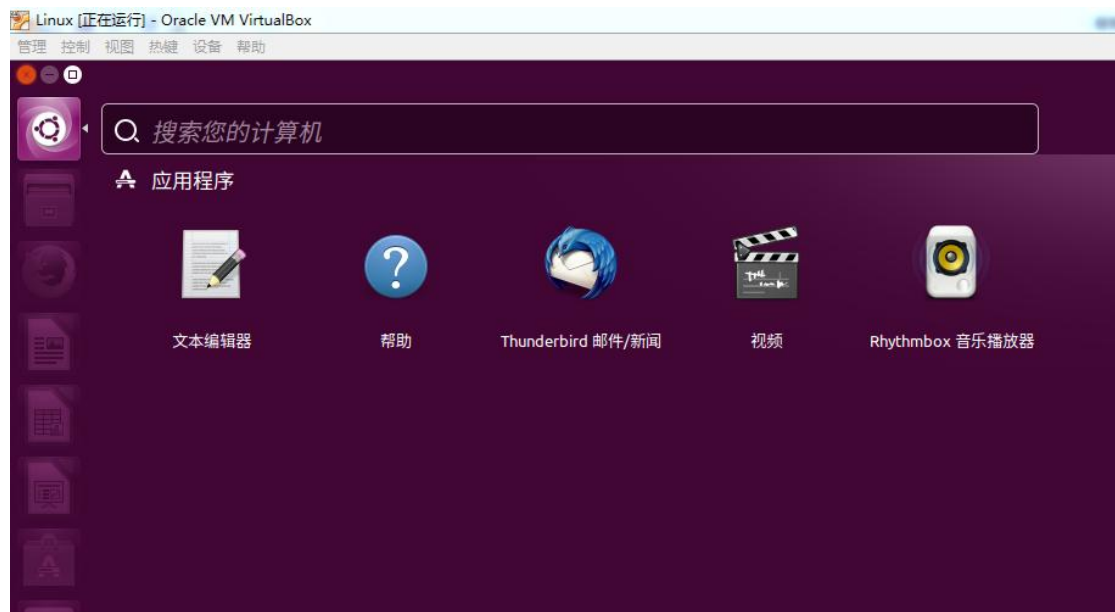
```
C:\GnuWin32\bin>lex1.yy.exe<1-1.cpp
103 17 5
```

三 在 LINUX 环境下的实验步骤

1. 首先安装 LINUX 虚拟机



2. 打开文本编辑器、



3. 输入 lex 文件

4. 存储文件

5. 输入源程序

6. 打开命令窗口

7. 运行程序

三 实验总结

通过本次实验，我学会了使用 Flex/Bison 构造编译器，利用 Flex 设计一个扫描器程序，用于计算一个文件中的字符数，单词数和行数。掌握了 Yacc 与 Lex 基本使用方法，此外还学习了在 LINUX 系统下完成该实验的方法。并且在老师的答疑下解决了一些重点难点，受益匪浅。