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：实验三 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

实验三 利用 Flex/Bison 构造编译器

一 实验目的

使用 bison 结合 flex 编写语法分析程序，对一段程序进行编译，并输出结

二 LEX 程序结构描述

```
%{
#include "Name.tab.h"          加载 Name.tab.h
#include <stdio.h>             加载 stdio.h
%}
char [A-Za-z]                  定义 char 为字符 A-Za-z
num [0-9]                      定义 num 为数字 0-9
eq [=]                         定义 eq 为等号
name {char}+                   定义 name 为字符串、若干个字符组合
age {num}+                     定义 age 为数值、若干个数字组合
%%
{name} { yylval = strdup(yytext); return NAME; }
                               若匹配到 name，将字符串拷贝到 yylval，
                               并返回 NAME
{eq} { return EQ; }            若匹配到 eq，返回 EQ
{age} { yylval = strdup(yytext); return AGE; }
                               若匹配到 age，将数值拷贝到 yylval，返回
                               AGE
%%
int yywrap()
```

{return 1;}

三　BISON 程序结构描述

```
%{
typedef char* string;
#define YYSTYPE string
%}
%token NAME EQ AGE
%%
file : record | record file
;
record : NAME EQ AGE {printf("%s is %s years old!!!\n", $1, $3);}
;
%%
int main(){
yyparse();return 0;
}
int yyerror(char *msg){
printf("Error encountered: %s \n", msg);
return 0;
}
```

YYSTYPE 定义了用来将值从 lexer 拷贝到解析器或者 Yacc 的 yylval（另一个 Yacc 变量）的类型

文件解析的语法

四 实验步骤

Windows 环境下安装 Bison

http://sourceforge.net/projects/gnuwin32/files/bison/2.4.1/

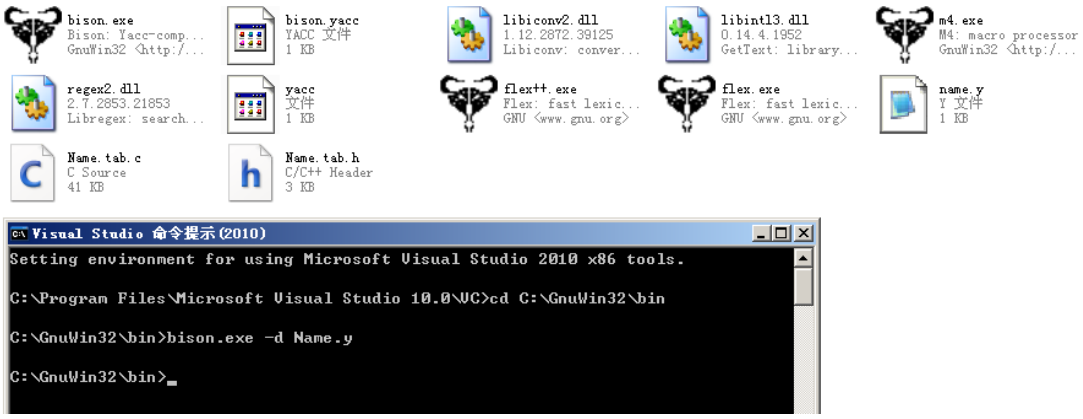
与 Flex 安装在同一目录，安装成功后。

编写源程序 Name.y，打开 Visual Studio 2008 命令行

进入 bison 安装目录 > cd C:\GnuWin32\bin

调用 bison.exe > bison.exe -d Name.y

生成 Name.tab.c， Name.tab.h



生成的 Name.tab.c， Name.tab.h

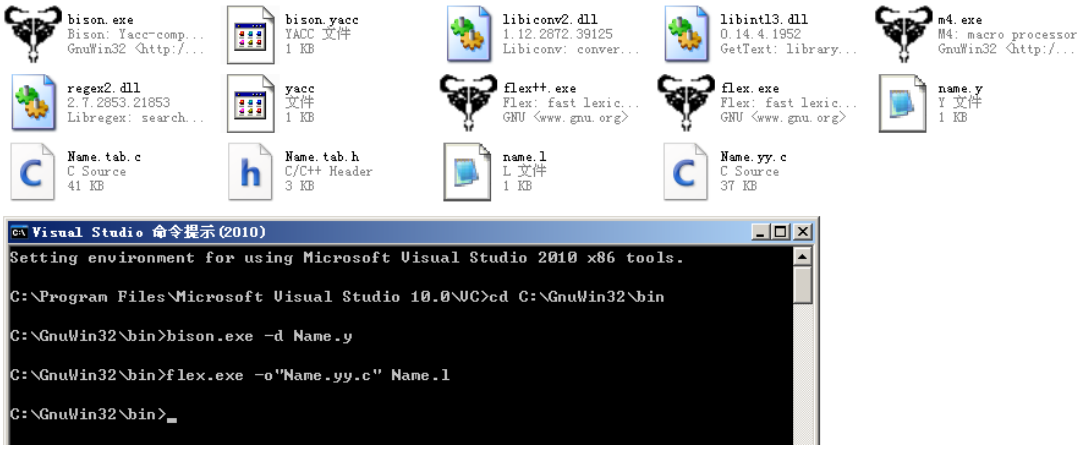
```
/* Tokens.  */
#ifndef YYTOKENTYPE
# define YYTOKENTYPE
   /* Put the tokens into the symbol table, so that GDB and other debuggers
      know about them.  */
   enum yytokentype {
     NAME = 258,
     EQ = 259,
     AGE = 260
   };
#endif


#if ! defined YYSTYPE && ! defined YYSTYPE_IS_DECLARED
typedef int YYSTYPE;
# define YYSTYPE_IS_TRIVIAL 1
# define yystype YYSTYPE /* obsolescent; will be withdrawn */
# define YYSTYPE_IS_DECLARED 1
#endif

extern YYSTYPE yylval;
```

编写源程序 Name.l ，打开 Visual Studio 2008 命令行

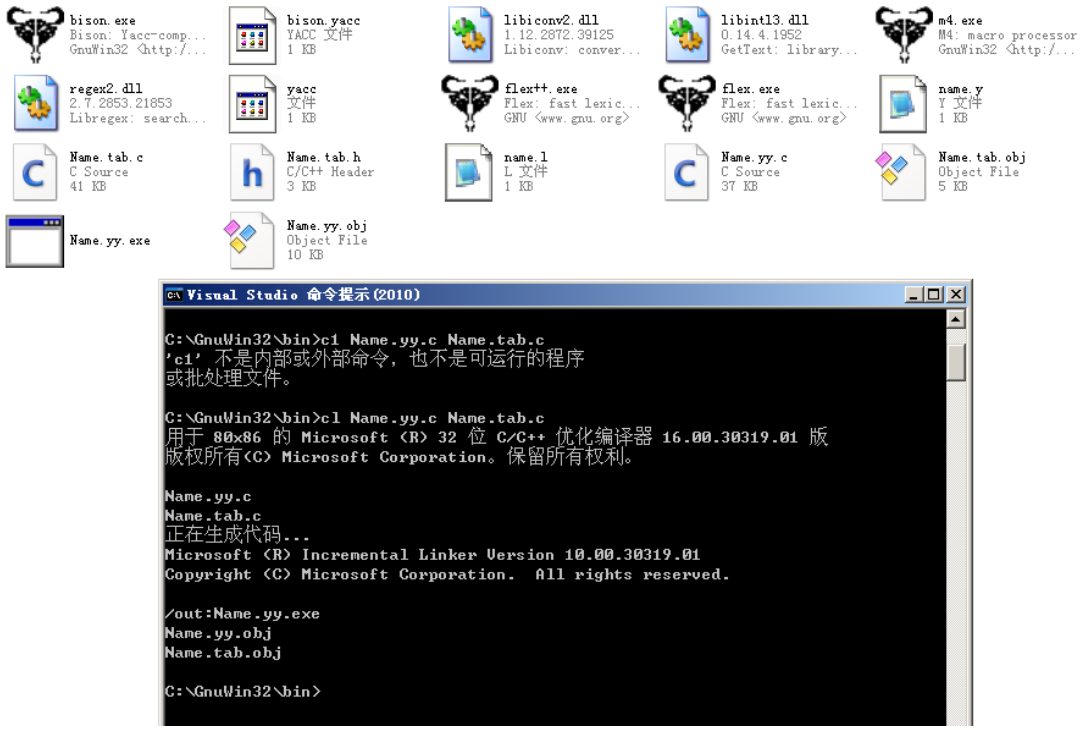
进入 flex 安装目录 > cd C:\GnuWin32\bin

调用 flex.exe > flex.exe -o"Name.yy.c" Name.l

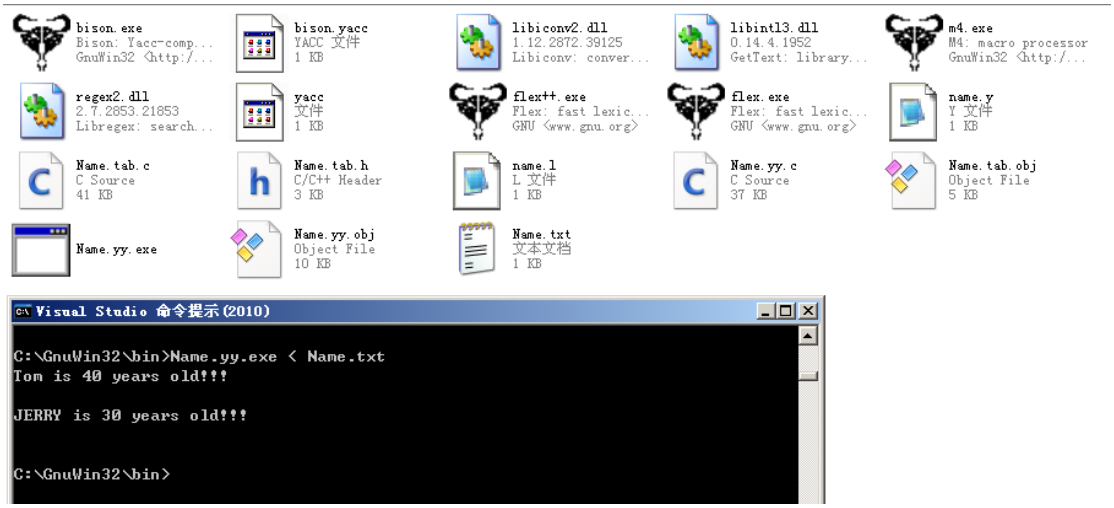
生成 Name.yy.c



调用 VS2008 编译器 cl.exe



调用 Name.yy.exe



五 实验总结

1. 你在编程过程中遇到了哪些难题？你是怎么克服的？

   在实验过程中，对于新接触的 BISON 源代码不是特别理解。

   我先是自己上网搜索 BISON 相关的语法资料，后来通过老师的帮助，顺利解决了所有的困难，完成了实验。

2. 你对你的程序的评价？

   本次实验完成的程序是一个简单的语法和语法分析器，和之前的实验相比增加了语法分析的环节，我认为在语法分析器的编程上，还有很多东西要学习与实践。

3. 你的收获有哪些

   通过本次实验，我先是编写 BISON 和 LEX 源文件，实现 C 语言子集的词法分析功能以及语法分析功能，然后编写一个测试程序，以给定的测试文件作为输入，输出运行结果到输出文件中。

   我进一步掌握了 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。

4. FLEX 与 BISON 的关系

通俗的说 flex 处理分词，将输出切分成一段一段的 token，这些 tokens 可以交给 Bison 处理，当然了，我们完全也可以不交给 bison，直接自己处理。

我们看到 Lex 从输入序列中识别标记。 如果你在查看标记序列，你可能想在这一序列出现时执行某一动作。 这种情况下有效序列的规范称为语法。 Yacc 语法文件包括这一语法规范。 它还包含了序列匹配时你想要做的事。 为了更加说清这一概念，让我们以英语为例。 这一套标记可能是：名词, 动词, 形容词等等。为了使用这些标记造一个语法正确的句子，你的结构必须符合一定的规则。 一个简单的句子可能是名词+动词或者名词+动词+名词。(如 I care. See spot run.)所以在我们这里，标记本身来自语言（Lex），并且标记序列允许用 Yacc 来指定这些标记(标记序列也叫语法)。

以本次实验为例，通过语法分析找出了“Tom=40 JERRY=30”这两个 record 中的 tokens，即 Tom JERRY 为 NAME，=为 EQ，30 40 为 AGE，完成了语法分析。然后在词法分析作出相应的输出。