**编译原理\_实验一**

班级：22计算机科学与技术3班

姓名：张雅瑞 学号：2022334323029

实验一 TINY语言的词法分析

1. 实验目的

构造tiny语言的词法分析器（扫描器），要求利用第三方的lex工具进行构造。实验结果：构造出的扫描器，能够读入教材样例中给出的tiny语言的示例代码，分解成token输出。

输入：

**{ Sample program**

**in TINY language -**

**computes factorial**

**}**

**read x; { input an integer }**

**if 0 < x then { don't compute if x <= 0 }**

**fact := 1;**

**repeat**

**fact := fact \* x;**

**x := x - 1**

**until x = 0;**

**write fact { output factorial of x }**

**end**

1. 实验设计

实验设计：安装flex，编写lex程序，使用flex将其编译为c程序，使用gcc将c程序编译为可执行文件（扫描器）。

输入：tiny语言的示例代码的文件名。

输出：分解的token。

相关正则表达式

数字标识符：[0-9]+

字符串标识符：[a-zA-Z]+

关键字标识符：read|if|then|repeat|until|write|end

操作符标识符：":="|";"|"<"|">"|"="|"+"|"-"|"\*"|"/"|"%"

注释的处理方法：

%x COMMENT

%%

[\n]\*        {printf("\nline %d:\n",line); line++;}

"{"                     { BEGIN(COMMENT); }

<COMMENT>[^{}]\*         {

                            printf("COMMENT BEGIN\n");

                            for(int i = 0; i < strlen(yytext); i++){

                                if(yytext[i] == '\n'){

                                    printf("\nline %d:\n",line); line++;

                                }

                                else{

                                    printf("%c",yytext[i]);

                                }

                            }

                            //printf("COMMENT BEGIN{ %s } COMMENT END\n",yytext);

                            printf("COMMENT END\n");

                        }

<COMMENT>"}"            { BEGIN(INITIAL); }

%%

1. 内容和步骤

1代码：

%{

    //包含C头文件、定义全局变量、声明函数

    #include <stdio.h>

    #include <stdlib.h>

    #include <string.h>

    int line=2;

%}

%x COMMENT

%%

[\n]\*        {printf("\nline %d:\n",line); line++;}

"{"                     { BEGIN(COMMENT); }

<COMMENT>[^{}]\*         {

                            printf("COMMENT BEGIN\n");

                            for(int i = 0; i < strlen(yytext); i++){

                                if(yytext[i] == '\n'){

                                    printf("\nline %d:\n",line); line++;

                                }

                                else{

                                    printf("%c",yytext[i]);

                                }

                            }

                            //printf("COMMENT BEGIN{ %s } COMMENT END\n",yytext);

                            printf("COMMENT END\n");

                        }

<COMMENT>"}"            { BEGIN(INITIAL); }

"read"       printf("READ ");

"if"         printf("IF ");

"then"       printf("THEN ");

"else"       printf("ELSE ");

"repeat"     printf("REPEAT ");

"until"      printf("UNTIL ");

"write"      printf("WRITE ");

"end"        printf("END ");

":="         printf("ASSIGN ");

";"          printf("SEMICOLON\n");

"<"          printf("LESS\_THAN ");

">"          printf("GREATER\_THAN ");

"="          printf("EQUALS ");

"+"          printf("PLUS ");

"-"          printf("MINUS ");

"\*"          printf("MULTIPLY ");

"/"          printf("DIVIDE ");

"%"          printf("MODULO ");

[0-9]+       printf("[NUMBER, Value = %s] ",yytext);

[a-zA-Z]+    printf("[IDENTIFIER, Name = %s] ",yytext);

[\t]         /\* ignore\*/

[ ]          /\* ignore\*/

.            printf("UNKNOWN ");

%%

int yywrap() {

    return 1;

}

int main(int argc, char \*argv[]) {

    if (argc != 2) {

        fprintf(stderr, "usage: %s filename\n", argv[0]);

        exit(1);

    }

    FILE \*file = fopen(argv[1], "r");

    if (!file) {

        fprintf(stderr, "could not open file: %s\n", argv[1]);

        exit(1);

    }

    yyin = file;

    printf("line1:\n");

    yylex();

    fclose(file);

    return 0;

}

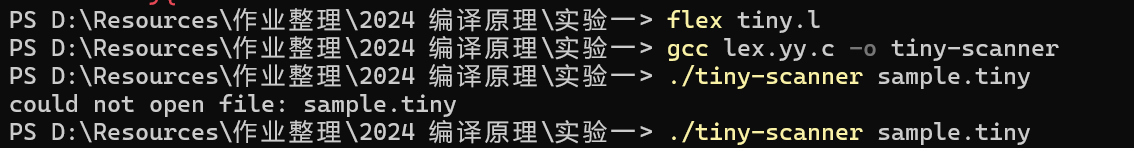
1. 结果：

编译命令：

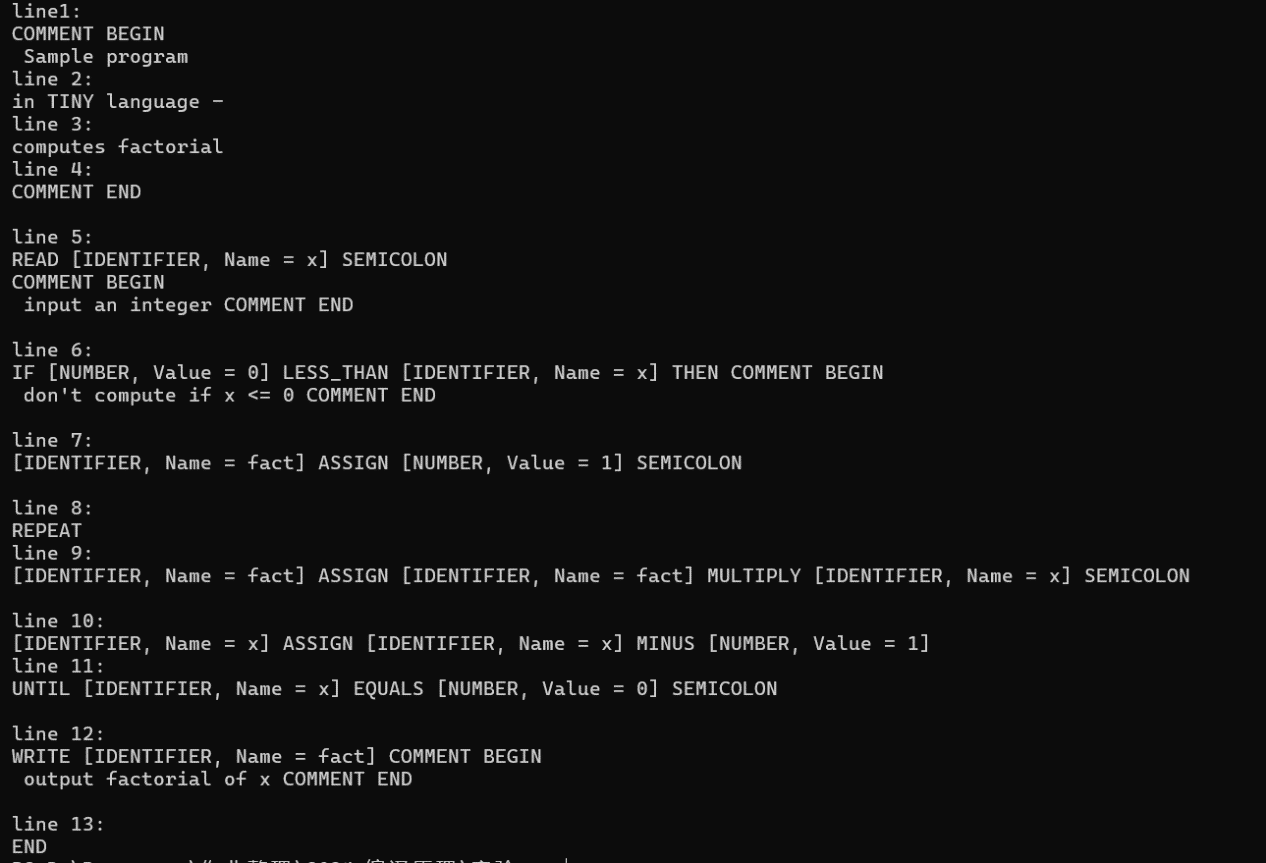
$ flex tiny.l

$ gcc lex.yy.c -o tiny-scanner

$ ./tiny-scanner sample.tiny #tiny语言示例代码文件



程序输出：



1. 实验结论:

**1. 理论基础评价**

在本次实验中，我对于构造词法分析器（扫描器）的理论知识掌握得非常清楚。我深入理解了词法分析在编译器设计中的重要性，并熟悉lex工具的工作原理和使用方法。我的实验设计合理，能够准确地将理论知识应用到实际的扫描器构造过程中，体现了扎实的理论基础和良好的应用能力。

**2. 分析和总结评价**

我对实验结果进行了完整和准确的描述。通过输入tiny语言的示例代码，我成功构造出了一个能够将其分解成token的扫描器。在实验过程中，我详细记录了每一步的操作和结果，包括lex程序的编写、flex的编译过程以及最终可执行文件的生成。此外，我还捕捉到了实验中的各种现象，如token的识别与输出等，并对其进行了准确的分析。我具备较强的信息综合能力，能够根据实验结果得出正确的结论，即成功构造出了tiny语言的词法分析器。

**3. 对工具的评价**

在本次实验中，lex工具及其对应的flex编译器发挥了关键作用。flex作为lex的替代品，具有更高的性能和更丰富的功能。其优点主要体现在以下几个方面：首先，flex支持更复杂的正则表达式，能够更准确地识别源代码中的token；其次，flex生成的扫描器代码具有较高的执行效率，能够满足实际应用的需求；最后，flex提供了良好的扩展性，方便用户根据需要进行定制和优化。

然而，flex也存在一些局限性和缺点。例如，flex的学习曲线较陡峭，对于初学者来说可能需要一定的时间来掌握其使用方法；此外，flex生成的扫描器代码可能较为庞大，对于某些小型项目来说可能显得过于复杂。针对这些局限性和缺点，我建议在使用flex时结合相关文档和教程进行学习，并根据实际情况进行优化和调整。

综上所述，通过本次实验，我成功利用lex工具构造出了一个能够对tiny语言进行词法分析的扫描器，展现了我的理论基础和实践能力。同时，我对实验过程和结果进行了准确的分析和总结，并对所使用的工具进行了客观的评价。实验结果符合预期目标，为今后的编译器设计和实现工作奠定了坚实的基础。