编译原理第一次实验报告

131220069 刚晋

测试环境

- Linux debian 3.2.0-4-686-pae
- flex 2.5.35
- bison (GNU Bison) 2.5
- gcc (Debian 4.7.2-5) 4.7.2

编译选项

完全使用了实验中原有提供的`Makefile`。仅为测试做出如下修改

```
--- Makefile.bak    2015-03-10 16:20:34.000000000 +0800
+++ Makefile    2016-03-30 02:12:21.000000000 +0800
@@ -34,7 +34,10 @@
 # 定义的一些伪目标
 .PHONY: clean test
 test:
-   ./parser ../Test/test1.cmm
+   @for TESTFILE in ../Test/*.cmm; do \
+       echo "========== $$TESTFILE ==========" ;\
+       ./parser $$TESTFILE ;\
+   done
 clean:
    rm -f parser lex.yy.c syntax.tab.c syntax.tab.h syntax.output
    rm -f $(OBJS) $(OBJS:.o=.d)
```

`make`位置为`./lab/Code/`

功能实现

词法分析

词法部分按照附录A中的规则书写。
选作部分为1.1，即识别8进制和16进制数。我讲识别非法8进制数和16进制数的工作放在词法分析中处理。
正则表达式为

```
oct         0[0-7]+
hex         0[xX][0-9a-fA-f]+
eoct        0[0-9]+
ehex        0[xX][0-9A-Za-z]+
```

语法分析

这部分工作基本完全按照讲义和附录实现。

语法树生成

树的节点结构定义如下

```
struct Node {
    struct Node *child[10];   //store childs
    int line;                       //store line NO.
    char type[64];                  //store Node type, including extra
information
    int val;                            //store num of childs, init as
0
};
```

暂时假定每个节点的子不超过10个。
语法树打印使用递归，包含两个参数`print(struct Node *root, int level)`，分别表示根节点和缩进等级。

错误恢复

我完成的错误恢复除讲义中提到的三个之外，还加入了一部分对于LB、LP、LC、SEMI等缺失的错误恢复。