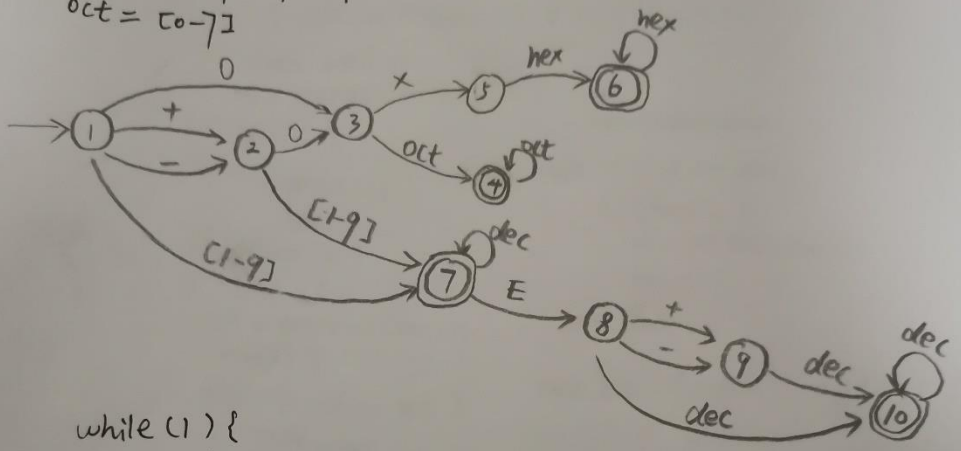
专业 网络工程 年级 17 班级 6 姓名 李斌 学号 20172320P2

一、

dec = [0-9]
hex = [0-9 a-f A-F]
oct = [0-7]




```
while(1){
    ch= getNextchar();
    state=1;
    switch(state){
      case 1: if(ch==0)
              { state=3;
                ch=getNextchar();
                break;}
            else if(ch=='+'||ch=='-')
            {  state=2;
               ch=getNextchar();
               break;}
            else if(ch==1||ch==2||.....ch==9)
            {  state=7;
               ch=getNextchar();
               break;
            }
            else{ error();
                  break;
                }
```

专业 网络工程 年级 17 班级 6 姓名 李斌 学号 20172132042

```
case 2: if(ch==0)
        { state=3;
          ch=getNextChar();
          break;}
        else if(ch==1 || ch==2 || ... || ch==9)
        { state=7;
          ch=getNextChar();
          break;}
        else{ error();}

case 3: if(ch=='x')
        { state=5;
          ch=getNextChar();
          break;}
        else if(ch==oct)
        { state=4;
          ch=getNextChar();
          break;}
        else error();

case4:  while(ch==oct)
          ch=getNextChar();
        Done();

case 5: if(ch==hex)
        { state=6;
          ch=getNextChar()
          break;}
        else error();

case 6: while(ch==hex)
          ch=getNextChar();
        Done();

case 7: if(ch=='E')
        { state=8;
          ch=getNextChar()
          break;}
        else if(ch==dec)
        { state=7;
          ch=getNextChar();
          break;}
        else Done();

case 8: if(ch=='+'||ch=='-')
        { state=9;
          ch=getNextChar();
          break;}
        else if(ch==dec)
        { state=10;
          ch=getNextChar();
          break;
        }
        else error();

case 9: if(ch==dec)
        { state=10;
          ch=getNextChar();
          break;
        }
        else error();

case 10: while(ch==dec)
           ch=getNextChar();
         Done();
}// end_switch
}// end_while
```

学号 专业 网络工程 年级 17 班级 6 姓名 [illegible]

二、DFA用邻接表来存储.

```
Struct .DFA {
    int a[max] ;//存来存终止状态
    vertex vertexs[max];
    int ver_numbers;  //点数量.
}nfa;

DfaTo Parsing Program().
    Edge *tmp ← NULL
    Gen("while(){")
    Gen("State=1;")
    Gen("Switch(State){")
    for i ← 0 to nfa.ver_number -1
        Gen("case"+i+":")
        tmp ← nfa.vertexs[i].Edge;
        if tmp != NULL
            Gen("if(ch=="+tmp.value+"){")
            Gen("State = "+tmp->end+";")
            Gen("ch=getNextChar();")
            Gen("break;")
            tmp ← tmp->nextEdge;
        while tmp != NULL
            Gen("else if(ch=="+tmp->value+"){")
            Gen("State="+tmp->end+";")
            Gen("ch=getNextChar();")
            Gen("break;")
            emp ← tmp->NextEdge
    if i in dfa.a
        Gen("else Done();")
    else Gen("else error();")
    //end_for
    Gen("}//endswitch")
    Gen("}//end_while")
```

专业 网络工程 年级 17 班级 6 姓名 李斌 学号 2017213204?

三、 ~~E → E~~

(1) 文法规则：

E → E || T | T

T → T && F | F

F → ~F | (E) | i rop i | ~~exp rop exp~~ | i

rop → > | < | ==

~~C → (A) | i~~

(2) 定义数据结构.

~~exp → A+B | B | A−B~~

~~B → B*C | B/C | C~~

```
typedef enum { Expk, Opk } Nodekind.
typedef enum { AND, OR, NOT } Opkind.
typedef enum { NUMBER } Expkind.
~~struct T~~
typedef struct treeNode {
    treeNode *lchild, *rchild;
    Nodekind nodekind;
    Opkind opkind;
    Expkind expkind;
    int val;
    char *str;
}
```

(3) 修改文法，消除左递归和左公因子.

E → T { || T }

T → F { && F }

~~F → i (ε | rop i) | ~F | exp rop exp | (E)~~

F → i (ε | rop i) | ~F | (E)

first(T) = { ~, i, (}

first(F) = { ~, i, (}

```
TreeNode *E() {
    if (token == '~' || token == 'i' || token == '(')
    { tmp = T(); while (token == "||")
        { match("||"); T() }
        TreeNode *node
        node = new TreeNode
        node.str = "||"
        node.lchild = tmp
        node.rchild = T().
        return node;
    } else error()
}

TreeNode *T() {
    if (token == '~' || token == 'i' || token == '(')
    { tmp = F(); < TreeNode *tmp;
        while (token == "&&")
        { match("&&");
            treeNode *node
            node = new TreeNode
            node.str = "&&"
            F();
            node.lchild = tmp
            node.rChild =
            return node;
        }
    }
    else error();
}

TreeNode *F() { ~~if (toke~~
    TreeNode *tmp;
    if (token == '~')
    { match('~');
      tmp == F(); }
    else if (token == '(')
    { match('(');
      tmp = E();
      match(')'); }
    ~~else if (token == i~~
```

专业 网络工程 年级 17 班级 6 姓名 [illegible] 学号 201721320492

```
else if (token == i)
{  TreeNode *node;
   node = new TreeNode;
   node.value = i;
   match(i);
   if (token == '>' || token == '<' || token == "==")
   {      tmp = token;
          match(token);
          TreeNode * opnode;
          opnode = new TreeNode;
          opnode.str = tmp;
          opnode.lchild = node;
          opnode.rchild = token;   // 此时 token 为 i
          match(i);
          return opnode;
   }
   return node
}
```

四、

for-stmt → for (exp) stmt-sequence

该文法不属于LR(0)文法，因为存在了同时归约和移进的状态。是SLR(1)文法。因为归约项的follow和移进项无交集，且归约归约之间follow也无交集。

五、 $I \to I^{(1)} rop$

$I' \to I\, I^{(2)}$

$I \to I^{(1)} rop$ { patchback($I^{(1)}.TC$)
$I.FC = I^{(1)}.FC$
$I.val = rop.val$ }

$I' \to I\, I^{(2)}$ { $I'.FC$ = merge($I.FC$, $I^{(2)}.FC$)
$I'.TC = I^{(2)}.TC$
}

Type → int | char | real | bool

Identifier → identifier
| identifir , Identifier

Ident-stmt → Type Identifier ;

六、(1) 先写出扩充的文法，并添加原文法中。
在程序的存储结构中添加新的结构。
在写递归子程序前，应先修改文法。

(2) 词法分析需在 getToken 里添加新的 case，并且在相应的枚举类型里面添加新变量。
语法分析写之前应先修改文法，然后遇到非终结符函数调用。终结符用match()匹配。语义分析在此基础上添加对应的动作。