专业_____年级____ 班级_____ 姓名_____学号_____

1.

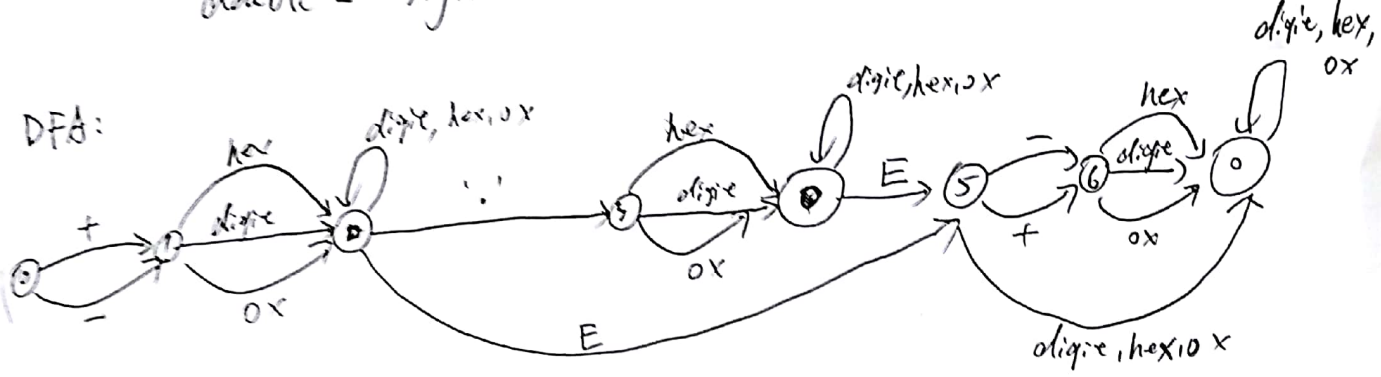
digit = [0-9] hex = [0-9 A-F a-f] 0x = [0-7]

nat = digit+ | hex+ | 0x+ c++ nat = digit+ | '0x' hex+ | '0' 0x+

signat = (+|-)? c++ nat

double = signat ('.' nat)? (E signat)?

DFA:




```
void
state = 0
switch (state)
case 0: input token
        if (token == '+' or '-')
        {state := 1
        } end case
case '1': input token
        if (token == digit | hex | 0x)
          state := 2
          end case
case '2': input token
        if (token == digit | hex | 0x)
          state := 2
        if (token == '.')
          state := 3
          end case
        if (token == E)
          state := 5
          end case
```

```
case '3': input token
        if token == (hex | digit | 0x)
          case state := 4
        end case
case '4': input token
        if (token == digit)
          state := 4
case '5': input token
        if token == ('-' or '+')
          state := 6
          end case.
case '6': input token
        if (token == digit)
          state := 8
        if (token == digit...)
          state := 7
case '7': input token
        if (token == digit |
          hex | 0x)
          @state :=
```

(共 页, 第 页)

二
五.1. 专业______ 年级____ 班级____ 姓名 黄纪东 ______ 学号______

// DFA图存储结构为邻接表.

```
class Node{
        int id;
        Node(){}
}
int size=100
class Edgelink{
        vector<pair<string,int>> edge; // <转换字符, 要转换后的状态>
}

class Graph{
        Edgelink Edge[size]
}

Void. DFA to Code(){
    for( i=1→size){
        cout<<"case"<< i <<":"<<endl;
        cout<<"{"switch (token)"<<endl;
        for (j=1→ edge.size())
        {
            cout<<"case<<j <<":"<<endl
            cout<<"state:=j"
            }

        cout <<"end case."<<endl
    }
    cout<<"end case"<<endl.

}
```

专业________ 年级____ 班级_____ 姓名 黄纲沐 学号 2017213110

三、设逻辑运算符优先级：!，&&、|| 由高到低，设 <,>,>=,<=,== 优先级高于&&和||.

1、

rop→ > | < | >= | <= | == |

E→ E&&T | T —改写→ E→ T{&& T}

T→ T||G | G —改写→ T → G{|| G}

P→ !E | G

G→ (E) | i rop i | E rop E

2、

```
struct BTreeNode {
    char data;
    vector<BTreeNode> child;
}
```

3、

```
BTreeNode* E()
{
    BTreeNode * temp ,temp2;
    temp =T();
    while (token=='&&' )
    {
        match('&&')
        temp2= new BTreeNode;
        temp2->data= '&&';
        temp2->child[0]=temp;
        temp2->child[1]= T();
        temp = temp2;
    }
    return temp;
}
```

其它类似

（共 页，第 页）

由 扫描全能王 扫描创建

专业______年级____ 班级_____ 姓名 吴兆海 学号 2017213010

四、(1) 新：statement → if-stmt | repeat-stmt | assign-stmt | read-stmt | write-stmt | for-stmt |

for-stmt → for (exp) stmt-sequence .

(2) programe → . stmt-sequence

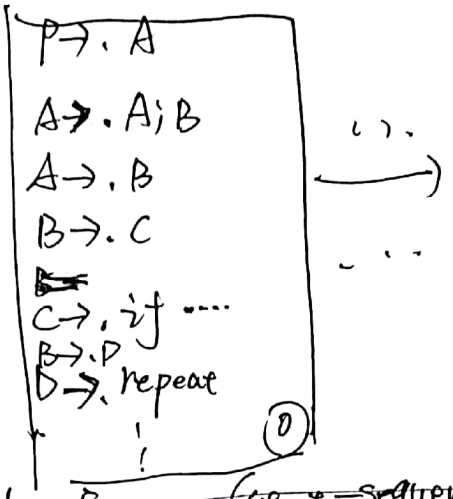
stmt-sequence → . stmt-sequence ; statement

文法化简
P → A
A → A ; B | B
B → C | D | E | F | H
C → if exp then A end | if exp than A else A end
= if exp than A [else A] end.

其它不变.

专业_____年级____ 班级_____ 姓名 [illegible] 学号 [illegible]

四、2.



~~follow First (stmt-sequence)={if, repeat, identifier; read,~~

DFA太复杂了画不完了.

画出DFA后看有没有移进-归约冲突，归约-归约冲突

若有则不是LR(0)文法. 反之为LR(0).

① 然后看有归约归约冲突则求两个归约产生式的follow

若follow的交集为空，则为SLR(0)文法.

② 如果有移进-归约冲突，归约项的follow与移进项的

First有交集则不为SLR(0).

反之为SLR(0)文法.

闽工 2017 6 吴维洋 201721310110.

五、

规则 (1) E→E || T | T

(2) T→T && G | G

(3) G→ ! E | P | (E)

(4) P→ i rop i | i rop i.

(4) {
P.FC.nextstate.
GEN [ ] nrop , entry(i) , entry(i), 0)
}

(3) {
G.FC=T.FC
}

(2) { T.FC = G.FC
}

(1) { E.FC=T.FC.
}

}

~ ⑷