Midterm answer key:

1.1 ( b )

1.2 ( c )

1.3 ( a )

1.4 ( b )

1.5 ( b )

1.6 ( c )

1.7 ( c )

1.8 ( d )

1.9 ( c )

1.10 ( a )

1.11 ( a )

1.12 ( b )

1.13 ( a )

1.14 ( c )

1.15 ( c )

1.16 ( a)

2.a aaa | bbb | ccc | abc

2.b ([abc][abc][abc]) | (ab[abc]\* | [abc]\* c)

a

2.c NFA for a\* | aba

ε

ε

ε

ε

Final state

a

Initial state

ε

ε

b

a

3.

<Record> 🡪 structtoken idtoken { <declaration> }

<declaration> 🡪 <simple decl> | <simple decl> ; <declaration>

<simple decl> 🡪 <type> <idlist>

<type> 🡪 integertoken | floattoken

<idlist> 🡪 id | id , <idlist>

4. First(S) = {a, b, d}

 First(A) = {b, d, ε}

 First(B) = {b, ε}

 First(D) = {d, ε}

 Follow(S) = {$}

 Follow (A) = {a}

 Follow(B) = {a, d}

 Follow(D) = {a}

Parsing table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | a | b | d | $ |
| S | S🡪 A a | S🡪 A a | S🡪 A a |  |
| A | A🡪 B D | A🡪 B D | A🡪 B D |  |
| B | B🡪 ε | B🡪 b | B🡪 ε |  |
| D | D🡪 ε |  | D🡪 d |  |

5.

P🡪 E {if (E.val == E.numop) print (“OP\_EXP”); else print (“Not OP-EXP”);}

E🡪 E1 + E2 {E.val = E1.val + E2.val; E.numop = E1.numop + E2.numop + 1;}

E🡪 E1 - E2 {E.val = E1.val - E2.val; E.numop = E1.numop + E2.numop + 1;}

E🡪 E1 \* E2 {E.val = E1.val \* E2.val; E.numop = E1.numop + E2.numop + 1;}

E🡪 E1 / E2 {E.val = E1.val / E2.val; E.numop = E1.numop + E2.numop + 1;}

E🡪 ( E1) { E.val = E1.val; E.numop = E1.numop;}

E🡪 num {E.val = num.val; E.numop = 0;}

Decorated parse tree for (1+2)\*2-1

print ("Not OP-EXP").

P

E

E.val = 5

E.numop = 3

E

E.val = 1

E.numop = 0

E

E.val = 6

E.numop = 2

E

E.val = 3

E.numop = 1

E

E.val = 3

E.numop = 1

E

E.val = 2

E.numop = 0

E

E.val = 1

E.numop = 0

num val=1

num val=2

num val=2

num val=1

E

E.val = 2

E.numop =0

+

)

-

(

\*