Name: Yichen Huang

CS403/503 Programming Languages
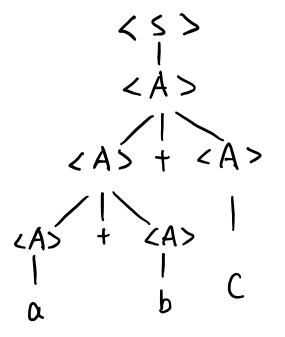**Spring 2021**
Assignment #1

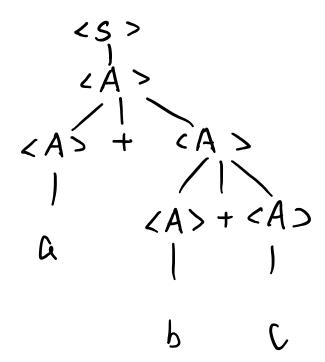1. Problem 3 of Chapter 3 on Page 157

Answer:

\<assign\> -> \<id\> = \<expr\>

\<id\> -> A | B | C

\<expr\> -> \<expr\> * \<term\>

      | \<term\>

\<term\> -> \<factor\> + \<term\>

      | \<factor\>

\<factor\> -> (\<expr\>) | \<id\>

2. Problem 8 of Chapter 3 on Page 158

Answer:

There could be two different parse tree which can prove the grammar is ambiguous

3. Problem 20 of Chapter 3 on Page 159

    3.2

    <assign> → <id> = <expr>

    <id> → A| B | C

    <expr> → <id> + <expr>

    | <id> * <expr>

    | (<expr>)

    | <id>

Answer:

- Syntax: <assign> -> <id> = <expr>

    Semantic : <expr>.expected_type <- <id>.actual_type

- Syntax: <expr>[1] -> <id> + <expr>[2]

    Semantic: <expr>[1].actual_type <-

        if (<id>.actual_type = int and <expr>[2].actual_type = int)

        then int

        else float

        end if

predicate: <expr>[1].actual_type == <expr>[1].expected_type

- Syntax: <expr>[1] -> <id> * <expr>[2]

  Semantic: <expr>[1].actual_type <-

    if (<id>.actual_type = int and <expr>[2].actual_type = int)

    then int

    else float

    end if

predicate: <expr>[1].actual_type == <expr>[1].expected_type

- Syntax: <expr>[1] -> (<expr>[2])

  Semantic: <expr>[1].actual_type <- <expr>[2].actual_type

- Syntax: <expr> -> <id>

  Semantic: <expr>.actual_type <- <id>.actual_type

- Syntax: <id> -> A | B | C

  Semantic: <id>.actual_type <- look-up(<id>.string)

4. Problem 8 of Chapter 4 on Page 194

Answer:

Name: Yichen Huang

Stack	Input	Action
0	(id + id) * id$	Shift 4
0(4	id + id) * id$	Shift 5
0(4id5	+ id) * id$	Reduce 6 , Goto[4,F]
0(4F3	+ id) * id$	Reduce 4, Goto[4,T]
0(4T2	+ id) * id$	Reduce 2, Goto[4,E]
0(4E8	+ id) * id$	Shift 6
0(4E8+6	id) * id$	Shift 5
0(4E8+6id5	) * id$	Reduce 6, Goto[6,F]
0(4E8+6F3	) * id$	Reduce 4, Goto[6,T]
0(4E8+6T9	) * id$	Reduce 1, Goto[4,E]
0(4E8	) * id$	Shift 11
0(4E8)11	* id$	Reduce 5, Goto[0,F]
0F3	* id$	Reduce 4, Goto[0,T]
0T2	* id$	Shift 7
0T2 * 7	id$	Shift 5
0T2 * 7id5	$	Reduce 6, Goto[7,F]
0T2 * 7F10	$	Reduce 3, Goto[0,T]
0T2	$	Reduce 2, Goto[0,E]
0E1	$	Accept