# 1. Section 1

Grammer

$$\begin{aligned} &< exp > ::= < exp > + < exp > \\ &< exp > ::= < exp > * < exp > \\ &< exp > ::= (< exp >) \\ &< exp > ::= id \end{aligned} \tag{1.1}$$

## 1.1 a

What is the set of tokens of the grammar?
Answer: id, +, *, (, )

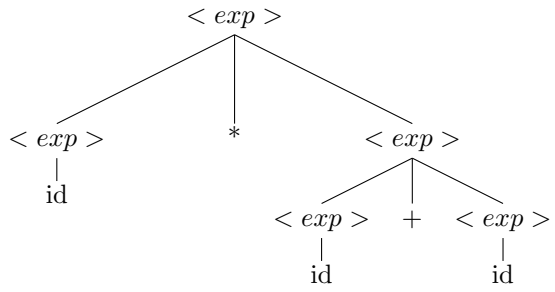
## 1.2 b

Write a leftmost derivation for id * id + id
Answer:

$$\begin{aligned} &< exp > \\ &< exp > * < exp > \\ &id * < exp > \\ &id * < exp > + < exp > \\ &id * id + < exp > \\ &id * id + id \end{aligned} \tag{1.2}$$

## 1.3 c

Draw a parse tree for id * id + id



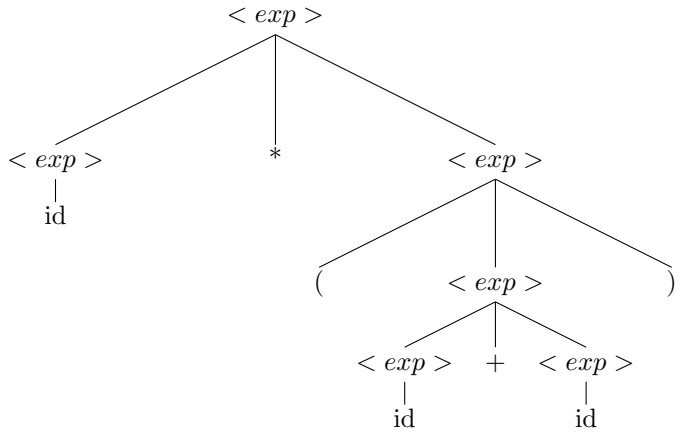

## 1.4 d

Write a rightmost derivation for id * ( id + id )
Answer:

$$
\begin{aligned}
&< exp > \\
&< exp > * < exp > \\
&< exp > *(< exp >) \\
&< exp > *(< exp > + < exp >) \\
&< exp > *(< exp + id) \\
&< exp > *(id + id) \\
&id * (id + id)
\end{aligned}
\tag{1.3}
$$

## 1.5 e

Drawn a parse tree for id * ( id + id )



# 2. Section 2

## 2.1 a

Write a grammar for the language of arithmetic expressions.

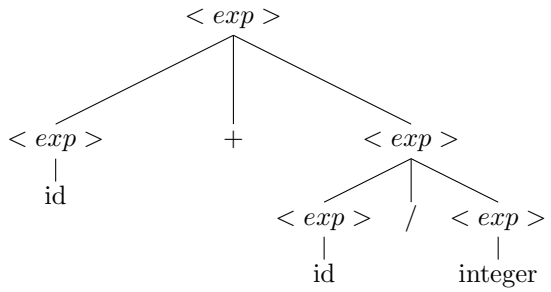
$$
\begin{aligned}
&< exp > ::= < exp > + < exp > \\
&< exp > ::= < exp > - < exp > \\
&< exp > ::= < exp > * < exp > \\
&< exp > ::= < exp > / < exp > \\
&< exp > ::= (< exp >) \\
&< exp > ::= id|integer|float
\end{aligned}
\tag{2.1}
$$

## 2.2 b

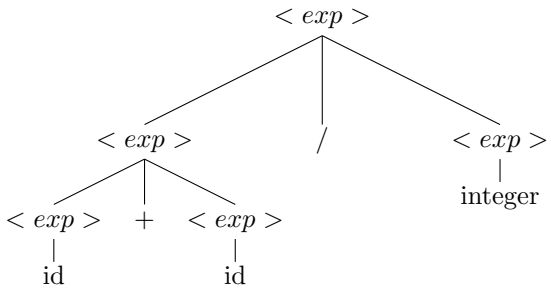
Draw a parse tree for expression id + id / integer using the grammar.

Answer: Two possible Solution

Solution 1:




Solution 2:

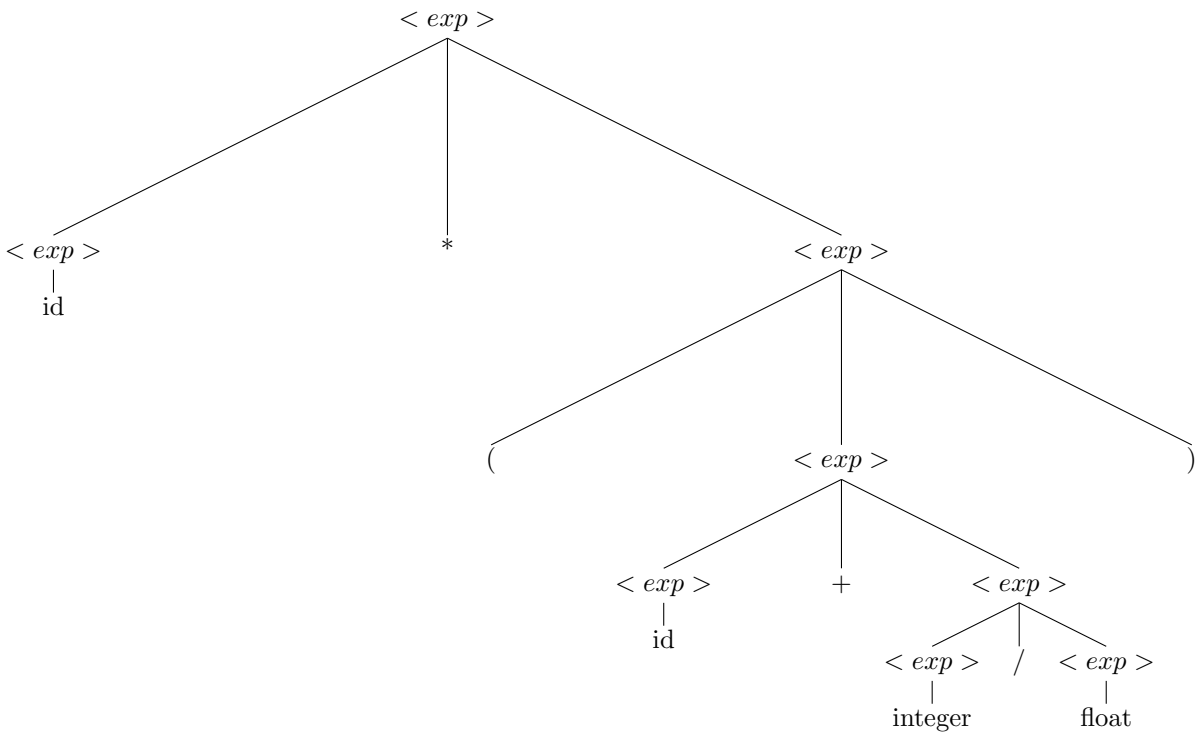



## 2.3 c

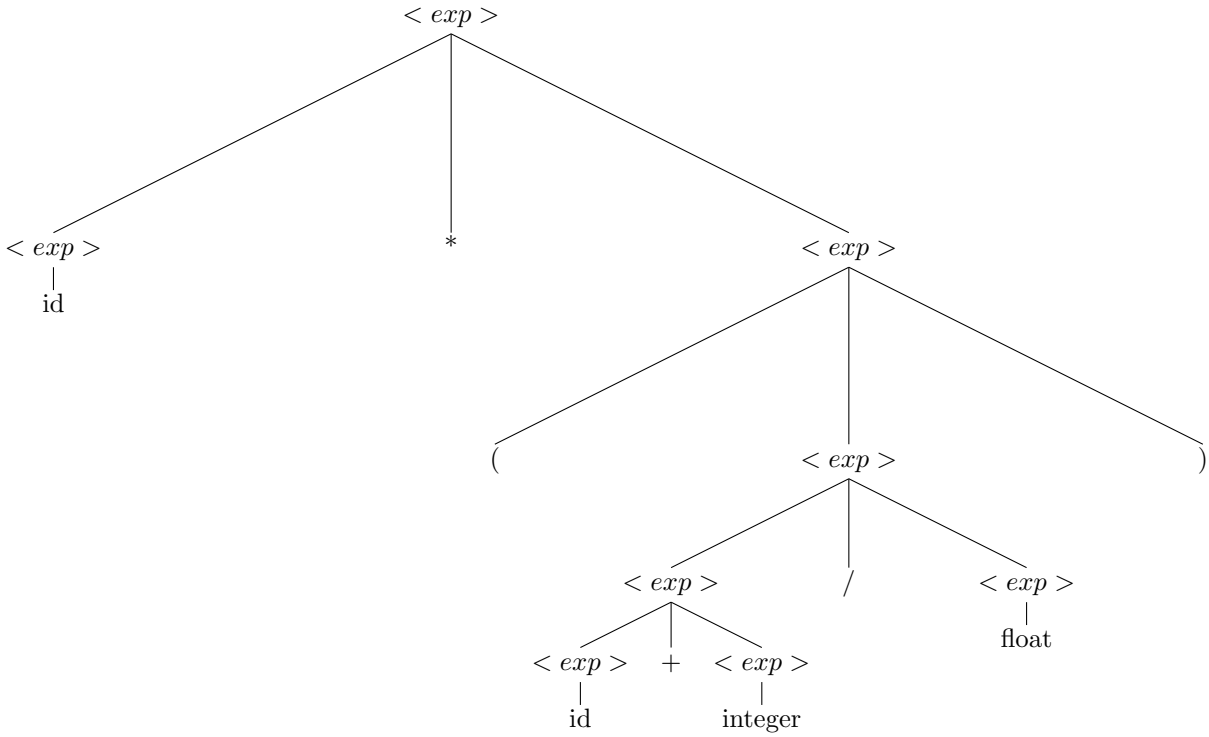
Draw a parse tree for expression id * ( id + integer / float ) using the grammar.

Answer: Two possible Solutions

Solution 1:


< exp >
< exp >
id
*
< exp >
(
< exp >
)
< exp >
+
< exp >
id
< exp >
/
< exp >
integer
float

Solution 2:


< exp >
< exp >
id
*
< exp >
(
< exp >
)
< exp >
/
< exp >
float
< exp >
+
< exp >
id
integer