CSCI 3155, Spring 2018 Lab 2 Writeup (02/09/2018)
Blaise Page blpa1200@colorado.edu
Justin Cai juca7186@colorado.edu

Problem 2

(a) The sentence generated by this language would start with one or more 'a's, followed by some number of 'b's and 'c's. It is important to note that every 'b' will have a corresponding 'c' because everytime the bBc case is invoked, it results in a 'b' and 'c' pair. Moreover, the sentence will also end with one or more 'a's.

(b) The following sentences are generated by the grammar:

1. baab

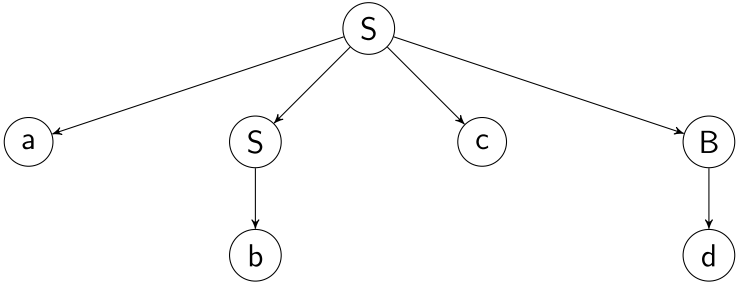
```
S => AaBb
  => baBb
  => baab
```

4. bbaab

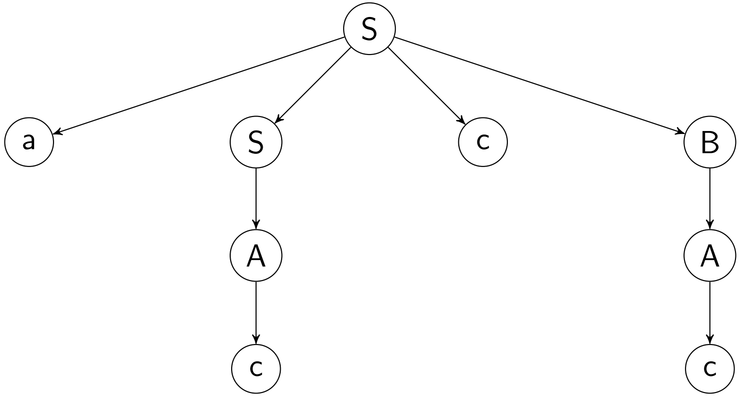
```
S => AaBb
  => AbaBb
  => bbaBb
  => bbaab
```

(c) The following sentences are generated by the grammar:

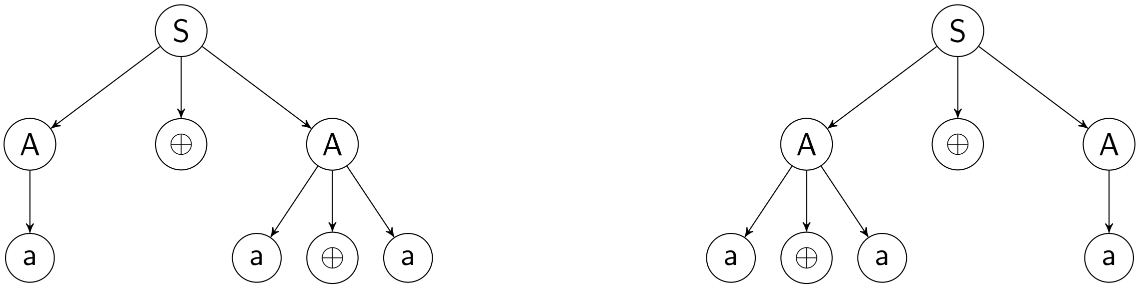
1. abcd



5. accc



(d) The following trees show that the grammar is ambiguous, because they are different trees, yet they generate the same sentence:



(d) The inference rules that inductively define the judgement form are as follows:

$$\frac{x \Downarrow n \quad y \Downarrow m}{x \oplus y \ \Downarrow\ n + m} \qquad \frac{}{a \ \Downarrow\ 1} \qquad \frac{}{b \ \Downarrow\ 0}$$

Problem 3

(a) (i) The expressions generated by the grammars consist of a combination of operands and operators, where the operands are seperated by operators.

(ii) Yes, the grammars generate the same expressions, but they do so with different parse trees.

(b) Our Scala expression for testing '-' or '<<' precedence is as follows:

```
if( 1 <<  3 - 2 ) == ( (1 << 3) - 2  )
  print "left shift has higher precedence"
else if ( 1 << 3 - 2 ) == ( 1 << (3 - 2) )
  print "minus has higher precedence"
```

The order in which the values are evaluated determines the precedence of the operators. Since scala evaluates its parse trees in a bottom-up fashion, the return from our expression shows the relative precedence of '-' and '<<'.

(c) The BNF grammar for floating points made up of fractions is as follows:

$$\begin{aligned}
floating\,point &::= (non\,zero\,integer)\text{‘.’}(fraction)\\
fraction &::= (middle\,sequence)|(fraction\,with\,exponent))\\
fraction\,with\,exponent &::= (non\,zero\,fraction)\text{‘E’}(non\,zero\,integer)\\
non\,zero\,fraction &::= (middle\,sequence)(non\,zero\,digit)\\
non\,zero\,integer &::= \text{‘-’}(non\,zero\,number)|non\,zero\,number\\
non\,zero\,number &::= (non\,zero\,digit)(middle\,sequence)\\
middle\,sequence &::= (any\,digit)(middle\,sequence)|\epsilon\\
any\,digit &::= \text{‘0’}|non\,zero\,digit\\
non\,zero\,digit &::= \text{‘1’}|\text{‘2’}|\text{‘3’}|\text{‘4’}|\text{‘5’}|\text{‘6’}|\text{‘7’}|\text{‘8’}|\text{‘9’}
\end{aligned}$$