Homework 4
CS 361

Ashley Vargas av11344n@pace.edu
Karina Vargas kv73396n@pace.edu

Grammars

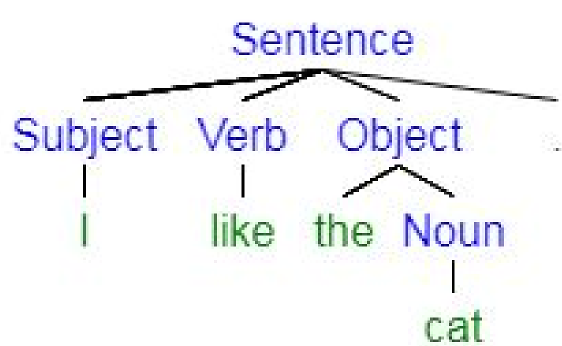
Exercise 1:

We consider the BNF grammar below:

Sentence	::=	Subject Verb Object .
Subject	::=	**I** | **a** Noun | **the** Noun
Object	::=	**me** | **a** Noun | **the** Noun
Noun	::=	**cat** | **mat** | **rat**
Verb	::=	**like** | **is** | **see** | **sees**

a. Show that **I like the cat.** is recognized by this BNF grammar using a rightmost derivation and, then, a parse tree.

Sentence = Subject Verb Object. -> Subject Verb the Noun. -> Subject Verb the cat. -> Subject like the cat. -> I like the cat.



b. Provide an expression that is NOT recognized by the grammar.

Rat bites cat.

Exercise 2:

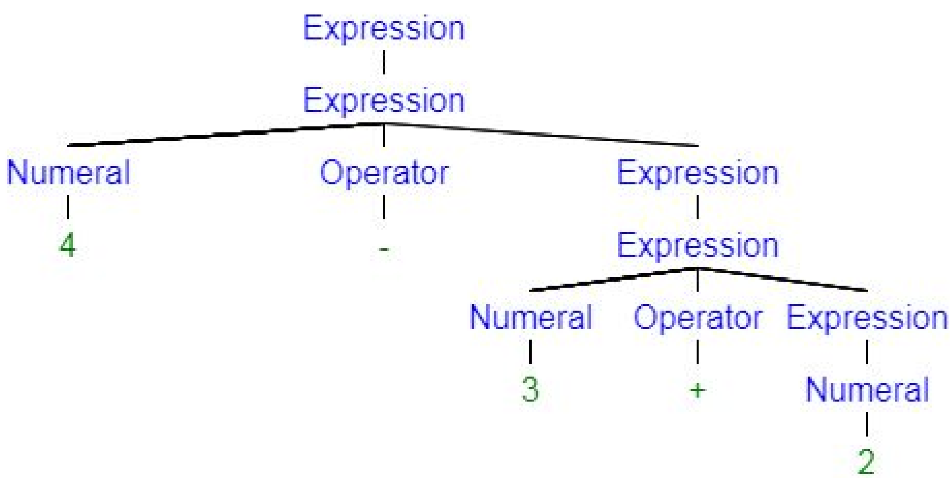
We consider the following grammar:

```
EXPRESSION ::= NUMERAL  |  ( EXPRESSION OPERATOR EXPRESSION
)
NUMERAL ::=  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

```
OPERATORS ::=  +  |  -
```

Show that (4 - (3 + 2)) is a legal EXPRESSION using a leftmost derivation, and,then, a parse tree.

Expression ::= (Expression Operator Expression) -> (Numeral Operator Expression) -> (4 Operator Expression) - > (4 - Expression) -> (4 - (Expression Operator Expression)) -> (4 - (Numeral Operator Expression)) ->(4 - (3 Operator Expression)) -> (4 - (3 + Expression)) ->(4 - (3 + Numeral)) -> (4 - (3 + 2))



Exercise 3:

Show that the following grammar is ambiguous:

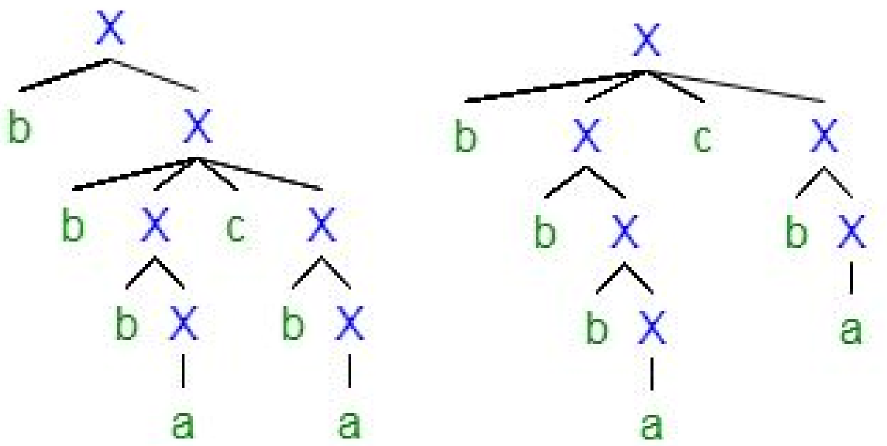
```
X -> a | bX | bXcX
where a,b,c are terminals.
```

You can make multiple parse trees from this grammar for one statement. For example, we can parse bbbacba in two different ways:

X -> bX -> bbXcX -> bbbXcX ->bbbacX -> bbbacbX ->bbbacba

X -> bXcX -> bbXcX -> bbbXcX -> bbbacX -> bbbacbX -> bbbacba

with this we could make two different parse trees.



Exercise 4:

a. Design a BNF grammar that recognizes expressions of the form Ai where A is in {a,b,c} and i is a digit.

Expression -> *Ai*
A -> *a* | *b* | *c* | *A* | *B* | *C*
Digit -> 0 | 1 | 2 | 3 | 4 | … | 9
i -> Digit

b. Design a BNF grammar that recognizes lists of the form A1, A2, A3, …, An. Use question a).

Expression -> *Ai*
List -> *Ai Separator Ai … Separator Ai*
A -> *a* | *b* | *c* | *A* | *B* | *C*
Digit -> 0 | 1 | 2 | 3 | 4 | … | 9
i -> Digit
Separator -> ,

Exercise 5:

1. Write a JAY program that computes the sum of the *n* first numbers with a loop.

void main () {
int sum;

```
int n;
int b = 1;

while(b<n+1 ) {
n = n +b;
sum = n;
b = b +1;
}

result = sum;
```

2. Write a JAY program that assigns the minimum of two numbers in a variable called min.

```
void main () {
int min;
int a;
int b;

if( a < b) {
min = a;
else min = b;
}
```

3. Provide 2 examples of lexical errors in JAY.

Two examples of lexical errors are:

```
3A
$$43~
```

4. Provide 2 examples of JAY programs with 2 different syntax errors.

Two examples of syntactical errors are:

```
3A = 4 + 2; //(3A is not an identifier)
if ( a > 3): //( it should be ; not :)
```

5. Provide 2 examples of JAY programs with errors that are neither detected during the lexical analysis nor during the syntactic analysis.

Two errors that are not detected are:

```
Area = w + h; //(should be * instead of +)
a1 == 2;  //(Should be = instead of == when assigning)
```

These are semantical errors.