Assignment 1
Due: Wednesday, 11 October 2017 at 11:59pm

This assignment asks you to prepare written answers to questions on BNF, EBNF, and Syntax Diagrams. Each of the questions has a short answer — no essays, please! You may discuss this assignment with other students and work on the problems together. However, *your write-up should be your own individual work. Please turn in your solutions electronically via Canvas by the due date.*

1. (15pts) Consider the following BNF grammar:

```
A ::= [ B , A ] | B
B ::= C | ( A ; C )
C ::= { C } | D
D ::= a | b | c
```

Note that in this grammar, `[`, `]`, `,`, `;`, `(`, `)`, {, }, `a`, `b`, and `c` are terminals, whereas `A`, `B`, `C`, and `D` are non-terminals. For each of the strings listed below, indicate *all* the non-terminals that can generate it (if there is none, write down "none"):

(a) (5pts) [c,(b;a)]

(b) (5pts) ([(a;b),c];a)

(c) (5pts) [(a;c),[(b;[a,c]),b]]

2. (20pts) Given the following BNF:

```
<integer> ::= <unsigned> | <sign> <unsigned>
<unsigned>::= <digits> | <unsigned><digits>
<digits> ::= <digits><digit> | <digit>
<digit> ::= 0 | 1 | ... | 9
<sign> ::= + | -
```

(a) (10pts) Convert the grammar into EBNF.

(b) (10pts) Show the syntax diagram of the above EBNF.

3. (25pts) What language is generated by each of the BNF grammars below (assuming that `<s>` is the start symbol):

(a) (5pts)

```
<s> ::= 0 <s> 1 1 1 1 | empty
```

(b) (10pts)

```
<s> ::= <x> | <y> | empty
<x> ::= 1 <x> 0 0 | empty
<y> ::= 0 <y> 1 | empty
```

(c) (10pts)

```
<s> ::= <x> | <y>
<x> ::= 0 <x> 1 | <x1>
<x1> ::= 0 <x1> | 0
<y> ::= 0 <y> 1 1 | <y1>
<y1> ::= <y1> 1 | 1
```

4. (20pts) Given the following grammar:

```
<stmt> ::= if <expr> then <stmt>
         | if <expr> then <stmt> else <stmt>
         | other
<expr> ::= true | false
```

where `other` is a terminal that stands for any other kinds of statements.

(a) (10pts) This grammar is ambiguous. Give a string having two different parse trees and draw the parse trees.

(b) (10pts) If we adopt the disambiguating rule (used in most languages) "match each else with the closest previous unmatched then," write an equivalent, unambiguous grammar.

5. (10pts) Given the following grammar:

```
<S> ::= if <E> then <S> else <S>
      | { <S> <D>
      | print <E>
<D> ::= } | ; <S> <D>
<E> ::= 0
```

Show the parse tree for the following string:

```
if 0 then { if 0 then print 0 else print 0 ; print 0 } else { print 0 }
```

6. (20pts) In a grammar, a grammar symbol X is *useless* if X can never appear in a derivation of a string.

(a) (15pts) Write an algorithm for eliminating all productions containing useless symbols from a given grammar.

(b) (5pts) Apply your algorithm to the following grammar:

```
<S> ::= 0 | <A> | <C>
<A> ::= <A> <B>
<B> ::= 1
<C> ::= 2
```

where `<S>` is the start symbol.