Context Free Grammar Exercises:

Exercise 1:

Consider the CFG G with non-terminals {E,T}, terminals {a, b,c,d,e}, start symbol E and productions:

 $E \rightarrow Eabc$

E→Ecde

 $E \rightarrow T$

T→bc

Transform G into an equivalent CFG G' without left recursion.

Solution:

Remember the rule we have seen in class:

<foo> →<foo>α₁</foo></foo>	Becomes:	$\langle foo \rangle \rightarrow \beta_1 \langle bar \rangle$
<foo> →<foo>α₂</foo></foo>		$< foo> \rightarrow \beta_1 < bar>$
$< foo > \rightarrow < foo > \alpha_n$		$\langle foo \rangle \rightarrow \beta_n \langle bar \rangle$
$\langle foo \rangle \rightarrow \beta_1$		
$\langle foo \rangle \rightarrow \beta_2$		$\langle bar \rangle \rightarrow \alpha_1 \langle bar \rangle$
		<bar> →α₂<bar></bar></bar>
$\langle foo \rangle \rightarrow \beta_n$		
		$\langle bar \rangle \rightarrow \alpha_n \langle bar \rangle$
		<bar>→ε</bar>

In the case of our grammar, we map E to foo, abc to α_1 , cde to α_2 , and T to β_1 . Therefore, we obtain G' as:

 $E \rightarrow TE'$

E'→abcE'

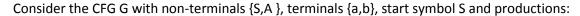
E'→cdeE'

E'**→**ε

T→bc

Note that E' is a new non-terminal we introduced to eliminate left-recursion.

Exercise 2:



 $S \rightarrow Aa$

S→b

A→Aab

A > ε

Is the grammar left-recursive? If yes, eliminate the left recursion.

Solution:

Yes, the grammar is left-recursive with productions $A \rightarrow Ab / \varepsilon$

Based on the rule we have seen in class, we map **A** to **foo**, **ab** to α_1 , **and** ε to β_1 . Therefore:

 $S \rightarrow Aa$

 $S \rightarrow b$

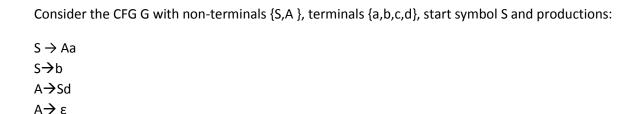
 $A \rightarrow A'$

A′→abA′

A′**→** ε

Note that A' is a new non-terminal we introduced to eliminate left-recursion.

Exercise 3:



Is the grammar left-recursive? If yes, eliminate the left recursion.

Solution:

Yes, the grammar is left-recursive since it can be re-written as follows:

 $S \rightarrow Aa$

 $S \rightarrow b$

A→Aad

A→bd

A→ ε

I have just replaced S by its value in $A \rightarrow Sd$. This shows that the grammar is clearly left-recursive with $A \rightarrow Aad \mid bd \mid \varepsilon$.

Based on the rule we have seen in class, we map **A** to **foo**, **ad** to α_1 , **bd** to α_2 . Therefore:

 $S \rightarrow Aa$

 $S \rightarrow b$

 $A \rightarrow A'$

 $A \rightarrow bdA'$

 $A' \rightarrow adA'$

A'**→** ε

Note that A' is a new non-terminal we introduced to eliminate left-recursion.

Exercise 4:



Solution:

Let's review the rule we have seen in class:

```
For each non-terminal A, find the longest prefix \alpha common to two or more of its alternatives If \alpha \neq \epsilon, then replace all of the A productions A \rightarrow \alpha \beta_1 | \alpha \beta_2 | \alpha \beta_3 | \dots | \alpha \beta_n With A \rightarrow \alpha A' A' \rightarrow \beta_1 | \beta_2 | \beta_3 | \dots | \beta_n
```

Hence, for **S >abcd |abc |ab**, the longest common prefix is **ab**. If we apply the transformation described in the rule above, we obtain:

```
S → abS'

| Tabc

S'→cd

| c

| ε
```

Furthermore, we notice that we can still perform left factoring on $S' \rightarrow cd/c$ since c is a common prefix.

Therefore, we obtain:

```
S \rightarrow abS'
\mid Tabc
S' \rightarrow cS''
\mid \epsilon
S'' \rightarrow d
\mid \epsilon
T \rightarrow e
```

Note that S' and S" are new non-terminals we introduced to perform left factoring.

Exercise 5:

Consider the CFG G with non-terminal {S,E}, terminals {if,then,else,a,b}, start symbol S and productions:

```
S→if E then S
| if E then S else S
| a
E→b
Perform left factoring on G.
```

Solution:

```
S→if E then SS'
| a
S'→else S
| ε
E→b
```

Exercise 6:

Given the grammar:

$$<$$
X $> ::= h | $\epsilon$$

Where <S>, <X>, and <Y> are the non-terminal symbols.

Find the FIRST sets for all non-terminals.

Solution:

We refer to the rules covered in class:

- 1) FIRST(terminal)is{terminal}
- 2) If $A \rightarrow a\alpha$, and a is a terminal:

{a}∈FIRST(A)

3) If $A \rightarrow B\alpha$, and rule $B \rightarrow \epsilon$ does **NOT** exist:

FIRST(B)∈FIRST(A)

4) If $A \rightarrow B\alpha$, and rule $B \rightarrow \epsilon$ **DOES** exist:

 $\{(FIRST(B) - \varepsilon) \cup FIRST(\alpha)\} \in FIRST(A)$

FIRST(X) =
$$\{h, \varepsilon\}$$
 (2nd rule)

FIRST(Y) =
$$\{p\}$$
 (2nd rule)

FIRST(S) = [FIRST(X)- ε] υ FIRST(Y) = {h, p} (4th rule)

Exercise 7:

Given the grammar:

Where <S>, <X>, and <Y> are the non-terminal symbols.

Find the FOLLOW sets for all non-terminals.

Solution:

We refer to the rules covered in class:

```
1){$}\epsilonFOLLOW($)
2) If A\rightarrow \alphaB:
```

 $FOLLOW(A) \in FOLLOW(B)$

3) If $A \rightarrow \alpha B \gamma$, and $\gamma \rightarrow \epsilon$ does **NOT** exist:

 $FIRST(\gamma) \in FOLLOW(B)$

4) If $A \rightarrow \alpha B \gamma$, and $\gamma \rightarrow \epsilon$ **DOES** exist:

 $\{(FIRST(\gamma)-\epsilon)\cup FOLLOW(A)\}\in FOLLOW(B)$

$$FOLLOW(X) = [FIRST(Y) - \varepsilon] \cup FOLLOW(S)$$

Exercise 8:

Given the grammar:

<S> ::= <X><Y>h

<X> ::= h | ε

<Y> ::= p | ε

Where <S>, <X>, and <Y> are the non-terminal symbols.

Find the FIRST and FOLLOW sets for all non-terminals.

Solution:

For the FIRST sets calculations:

FIRST(X)= $\{h, \epsilon\}$ (2nd rule ... or just common sense!)

FIRST(Y)= $\{p, \epsilon\}$ (2nd rule as well...)

Now, the FIRST(S) is the only challenging one.

I know that the 2nd rule will not apply since <S> derives a string that does not start with a terminal.

How about the 3rd rule? I will try to apply it by mapping the symbols of the 3rd rules onto the symbols of the only production where <S> appears on the left-hand side (remember, when you are finding the FIRST for a terminal, you only look at the productions where that terminal appears on the left-hand side).

The mapping works great. However, the 3^{rd} rule specifically says that $B \rightarrow \varepsilon$ should **NOT** exist. However, I do have a production $X \rightarrow \varepsilon$. Therefore, I cannot apply the 3^{rd} rule. So, let's analyze the 4^{th} rule (which has the same mapping as the 3^{rd} rule):

Mapping works great! Also, the 4th rule says that B \rightarrow ϵ should exist (which it does due to X \rightarrow ϵ). So, I apply the rule as follows:

[(FIRST(X)- ϵ) υ FIRST(Yh)] ϵ FIRST(S)

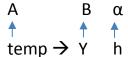
I know already what is the FIRST(X), but what about FIRST(Yh)? I do not have it yet and therefore, I need to calculate it. How do I calculate the FIRST of two symbols? That's simple, just create a production that summarizes the two symbols into one as follows:

temp→Yh

Therefore, if I find the FIRST(temp), I have the FIRST(Yh). Note that temp is just a temporary symbol I created to aid with this operation and will discard immediately after.

I know that I cannot find the FIRST(temp) using the 2^{nd} rule because temp derives a string that does not start with a terminal. So it's either the 3^{rd} or 4^{th} rule.

The 3rd rule maps quite well:



However, it is not applicable since $Y \rightarrow \epsilon$. But the 4th rule which has the same mapping works and so I apply it:

[(FIRST(Y)- ε) υ FIRST(h)] ε FIRST(temp)

I have the FIRST(Y). I also know that the FIRST(h) is simply {h} (see the 1st rule for the FIRST set calculation). Therefore:

FIRST(temp)={h,p}=FIRST(Yh)

Remember, I did all that work to arrive at the FIRST(S). I had the following relationship:

[(FIRST(X)- ε) υ FIRST(Yh)] ε FIRST(S)

Since, I now know the FIRST(X) and FIRST(Yh), I can populate the FIRST(S) set:

 $FIRST(S)=\{h,p\}$

For the FOLLOW sets calculation:

FOLLOW(S)= $\{\$\}$ (using the 1^{st} rule).

To obtain the FOLLOW(X), I need to analyze the following production (since it's the only one where X appears on the right-hand side):

The 2nd rule does not work since X does not appear at the end of the string XYh. However, the 3rd rules maps well as shown below:

A
$$\alpha B \gamma$$

$$\uparrow \uparrow \uparrow \downarrow \downarrow$$
S $\rightarrow \epsilon X Y h$

The 3rd rule states that $\gamma \rightarrow \epsilon$ does not exist. However, γ refers to two symbols instead of one. Therefore, the only way $\gamma \rightarrow \epsilon$ would exist is if $Y \rightarrow \epsilon$ and $h \rightarrow \epsilon$. While Y does derive epsilon, h cannot derive anything other than itself, since it is a terminal. Therefore, you will never have a situation where $\gamma \rightarrow \epsilon$. I can safely conclude that $\gamma \rightarrow \epsilon$ does not exist. Hence, the 3rd rule is applicable as follows:

 $FIRST(Yh) \in FOLLOW(X)$

We already have FIRST(Yh). Therefore:

 $FOLLOW(X)=\{h,p\}$

To obtain FOLLOW(Y), I also simply apply the 3rd rule which yields:

 $FIRST(h) \in FOLLOW(Y)$

This means that:

 $FOLLOW(Y)=\{h\}$