

CSE340 SPRING 2020 - HOMEWORK 2

Due: Wednesday February 2 2020 by 11:59 PM on Canvas

All submissions should be **typed**, no exceptions.

1. Consider the grammar.

$$S \rightarrow B A g \text{ (1)}$$

$$A \rightarrow f A b \text{ (2)}$$

$$A \rightarrow (C A B) \text{ (3)}$$

$$A \rightarrow C \text{ (4)}$$

$$B \rightarrow a B c \text{ (5)}$$

$$B \rightarrow \varepsilon \text{ (6)}$$

$$C \rightarrow c C \text{ (7)}$$

$$C \rightarrow \varepsilon \text{ (8)}$$

where S, A, B, C and D are the non-terminals, S is the start symbol, and a, b, c, f, g, '(', and ')' are the terminals.

1.1. **FIRST sets.** Do the following

1.1.1. Do an initialization pass by applying FIRST sets rules I and II. Show the resulting FIRST sets

$$\text{FIRST}(\varepsilon) = \{\varepsilon\}$$

$$\text{FIRST}(a) = \{a\}$$

$$\text{FIRST}(b) = \{b\}$$

$$\text{FIRST}(c) = \{c\}$$

$$\text{FIRST}(f) = \{f\}$$

$$\text{FIRST}(g) = \{g\}$$

$$\text{FIRST}(() = \{()$$

$$\text{FIRST}()) = \{) \}$$

1.1.2. Do one pass on the grammar rules in the order in which they are listed as follows. For each

grammar rules, apply FIRST set rule III, then apply FIRST set rule IV then apply FIRST set rule V. Show the resulting FIRST sets after this one pass.

$$\text{FIRST}(S) = \{ \}$$

$$\text{FIRST}(A) = \{f, (\}$$

$$\text{FIRST}(B) = \{a, \varepsilon\}$$

$$\text{FIRST}(C) = \{c, \varepsilon\}$$

1.1.3. Show the final result of the calculation of the FIRST sets

$$\text{FIRST}(S) = \{a, f, (, c, g\}$$

$$\text{FIRST}(A) = \{f, (, c, \varepsilon\}$$

$\text{FIRST}(B) = \{a, \epsilon\}$

$\text{FIRST}(C) = \{c, \epsilon\}$

1.2. **FOLLOW sets.** Do the following

1.2.1. Do an initialization pass by applying FOLLOW set rules I. Then do one pass by applying

FOLLOW sets rules IV, and V. Show the resulting FOLLOW sets

$\text{FOLLOW}(S) = \{\$ \}$

$\text{FOLLOW}(A) = \{ \}$

$\text{FOLLOW}(B) = \{f, (, c\}$

$\text{FOLLOW}(C) = \{ \}$

1.2.2. Do one pass on the grammar rules in the order in which they are listed as follows. For each

grammar rules, apply FOLLOW set rule II, then apply FOLLOW set rule III. Show the resulting FOLLOW sets after this one pass.

$\text{FOLLOW}(S) = \{\$ \}$

$\text{FOLLOW}(A) = \{g\}$

$\text{FOLLOW}(B) = \{f, (, c, g\}$

$\text{FOLLOW}(C) = \{ \}$

1.2.3. Show the final result of the calculation of the FOLLOW sets ‘

$\text{FOLLOW}(S) = \{\$ \}$

$\text{FOLLOW}(A) = \{g \ b \ a \)\}$

$\text{FOLLOW}(B) = \{f \ (\ c \ g \)\}$

$\text{FOLLOW}(C) = \{f \ (\ c \ a \) \ g \ b\}$

2. Consider the grammar

$S \rightarrow A B C$

$A \rightarrow a A b \mid c \mid B$

$B \rightarrow b B c \mid D$

$C \rightarrow b B c \mid$

ϵ

$D \rightarrow d \mid \epsilon$

Show that this grammar does not have a recursive descent predictive parser. To get full credit you should show all the conditions of predictive parsing that are not satisfied by this grammar. You will need to calculate FIRST and FOLLOW sets but you do not need to show how you calculated them. You will need to explicitly show which conditions of predictive parsing are not satisfied.

$\text{FIRST}(a) = \{a\}$

$\text{FIRST}(b) = \{b\}$

$\text{FIRST}(c) = \{c\}$

$\text{FIRST}(d) = \{d\}$
 $\text{FIRST}(\epsilon) = \{\epsilon\}$
 $\text{FIRST}(S) = \{a, c, b, d, \epsilon\}$
 $\text{FIRST}(A) = \{a, c, b, d, \epsilon\}$
 $\text{FIRST}(B) = \{b, d, \epsilon\}$
 $\text{FIRST}(C) = \{b, \epsilon\}$
 $\text{FIRST}(D) = \{d, \epsilon\}$

$\text{FOLLOW}(S) = \{\$ \}$
 $\text{FOLLOW}(A) = \{b, d, \$ \}$
 $\text{FOLLOW}(B) = \{b, \$, d, c\}$
 $\text{FOLLOW}(C) = \{\$ \}$
 $\text{FOLLOW}(D) = \{b, \$, d, c\}$

3. Consider the grammar

$S \rightarrow A b C \mid D c A$
 $A \rightarrow a A d \mid E \mid f$
 $E \rightarrow e \mid \epsilon$
 $C \rightarrow c \mid \epsilon$
 $D \rightarrow d D \mid \epsilon$

3.1. Show that the grammar has a predictive recursive descent parser. You should show that the conditions

of predictive parsing apply for every non-terminal.

3.2. Write `parse_S()`, `parse_A()`. Your parser should follow the general model of predictive parser that we saw in class. In particular, for non-terminals that can generate ϵ , the parser should check the FOLLOW set before choosing to parse the righthand side that generates ϵ .

3.3. Give a **full execution** trace for your parser from part 3.2 above on input `a a d d`.

4. We say that a symbol of a grammar is useless if it does not appear in the derivation of a strings of terminals. Formally, *A is useful* if there exists a derivation $S \Rightarrow^* x A y \Rightarrow^* w$ where w is a string of terminals. *A is useless* if it is not useful. Note that a useful symbol can be either a terminal or a non-terminal. Also, a useless symbol can be a terminal or a non-terminal.

Consider the grammar

$S \rightarrow A b C \mid D c A$
 $A \rightarrow a C$
 $E \rightarrow e D \mid \epsilon$
 $C \rightarrow c A$

$$D \rightarrow d D \mid E$$

Which are the useful symbols of this grammar? For every useful symbol, give a derivation that shows that the symbol is useful.

5. Consider the following operator grammar

$$E \rightarrow E \& E$$

$$E \rightarrow E ! E$$

$$E \rightarrow E ^ E$$

$$E \rightarrow \sim E$$

$$E \rightarrow (E) \mid id$$

& is a right associative binary operator. ! and ^ are left associative binary and have the same precedence. & has higher precedence than ! and ^. ~ is a unary operator and has lower precedence than & and higher precedence than ! and ^.

5.1. Draw the precedence table for this grammar 5.2.

Show step by step how $\sim id \& id ! id ^ id$ is parsed

Note. Refer to solution of homework from last semester for expected format of your answers.