### CS214 Proj02 Chan Kim

### Project 2: Practicing with BNFs

1. Using these ideas, define the constructs below with BNF productions. Underline the terminals in your productions, to distinguish them from the nonterminals. You may assume that the following have already been defined:

<letter> ::= A | B | C | D | E | F | G | H | I | J | K | L

| M | N | O | P | Q | R | S | T | U | V | W

| X | Y | Z | a | b | c | d | e | f | g | h

| i | j | k | l | m | n | o | p | q | r | s

| t | u | v | w | x | y | z

<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

* 1. A Java character literal  
     <*character literal*> ::= <letter> | <digit> | /t | /b | /n | /r | /f | /’ | /” | \\ | <character literal>
  2. A Java character string literal  
     <string *literal*> ::= <letter> | <digit> | <string literal><string literal> | Ø
  3. A Java integer literal  
     <hex> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7  
     <octal> ::= 0 | … | 9a | … | f  
     <hexs> ::= <hex><hexs> | Ø  
     <octals> ::= <octal><octals> | Ø  
     <digits> ::= <digit><digits> | Ø   
     <*integer literal*> ::= <*digits*> | <*hexs*> | 0x <octal*s*>
  4. A Java real (floating point) literal   
     <digits> ::= <digit><digits> | Ø   
     <*floating-point literal*> ::= <*digits*> . [ <*digits*> ]
  5. A Java identifier:   
     <identifier> ::= ( <letter> | $ | \_ ) { <identifier> | <digits> | Ø }
  6. A Java function declaration (prototype):   
     <modifier> ::= public | private  
     <method> ::= class<?> | int | boolean | String | Type | Object | Void  
     <type> ::= boolean | byte | char | short | int | float | long | double  
     <parameter> ::= <type> <identifier>, [ <parameter> ]  
       
     <function declaration> ::= <modifier> <method> <identifier> “(“ [ <parameter> ] “)” “{” <statement> “;” “}”
  7. A Java if statement (you may assume that the nonterminals <statement> and <expression> are defined elsewhere):   
     <if> ::= “if” “(“ <expression> “)” <statement> [ “else” <statement> ]
  8. A Java while statement (you may assume that the nonterminals <statement> and <expression> are defined elsewhere):   
     <while> ::= “while” “(“ <expression> “)” <statement>

1. Prove that the following grammar is ambiguous:

<S> ::= <A>  
<A> ::= <A> + <A> | <id>  
<id> ::= a | b | c

|  |  |
| --- | --- |
| <S> <A> <A> + <A> <A> + <A> <id> <id> <id> c  a b | <S> <A> <A> + <A> <id> <A> + <A>  a <id> <id>  b c |

1. Give a left-most derivation for A = A \* (B + C) using the following BNF grammar:

<assign> ::= <id> = <expr>  
<id> ::= A | B | C  
<expr> ::= <expr> + <term> | <term>  
<term> ::= <term> \* <factor> | <factor>  
<factor> ::= ( <expr> ) | <id>

<assign>  
<id> = <expr>  
 A <term>  
 <term> \* <factor>  
 <factor> ( <expr> )  
 <id> ( <expr> + <term> )  
 A <term> <factor>   
 <factor> <id>  
 <id> C  
 B