CprE 342
Homework 1

Sean Gordon

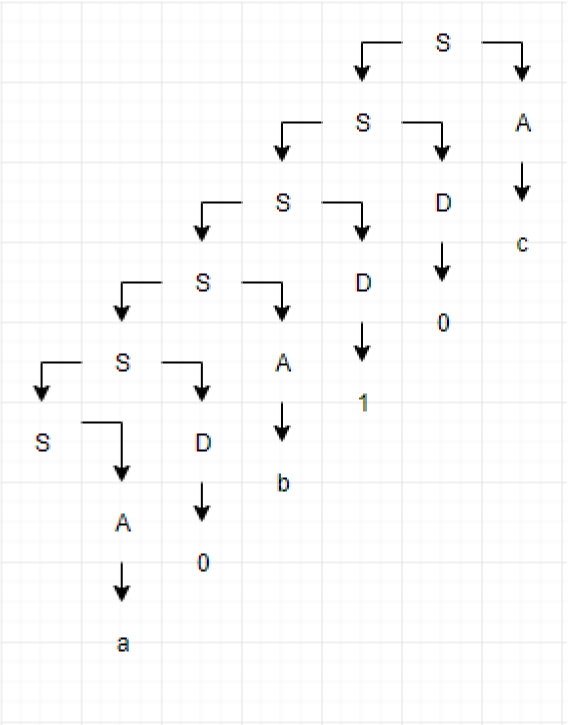
1a)

Terminals	**Non-Terminals**
• a • c • 1 • b • 0	• S • A • D

b) S ⇒ SA ⇒ SDA ⇒ SDDA ⇒ SADDA ⇒ SDADDA ⇒ ADADDA ⇒ aDDADA ⇒ a0ADDA ⇒ a0bDDA ⇒ a0b1DA ⇒ a0b10A ⇒ a0b10c

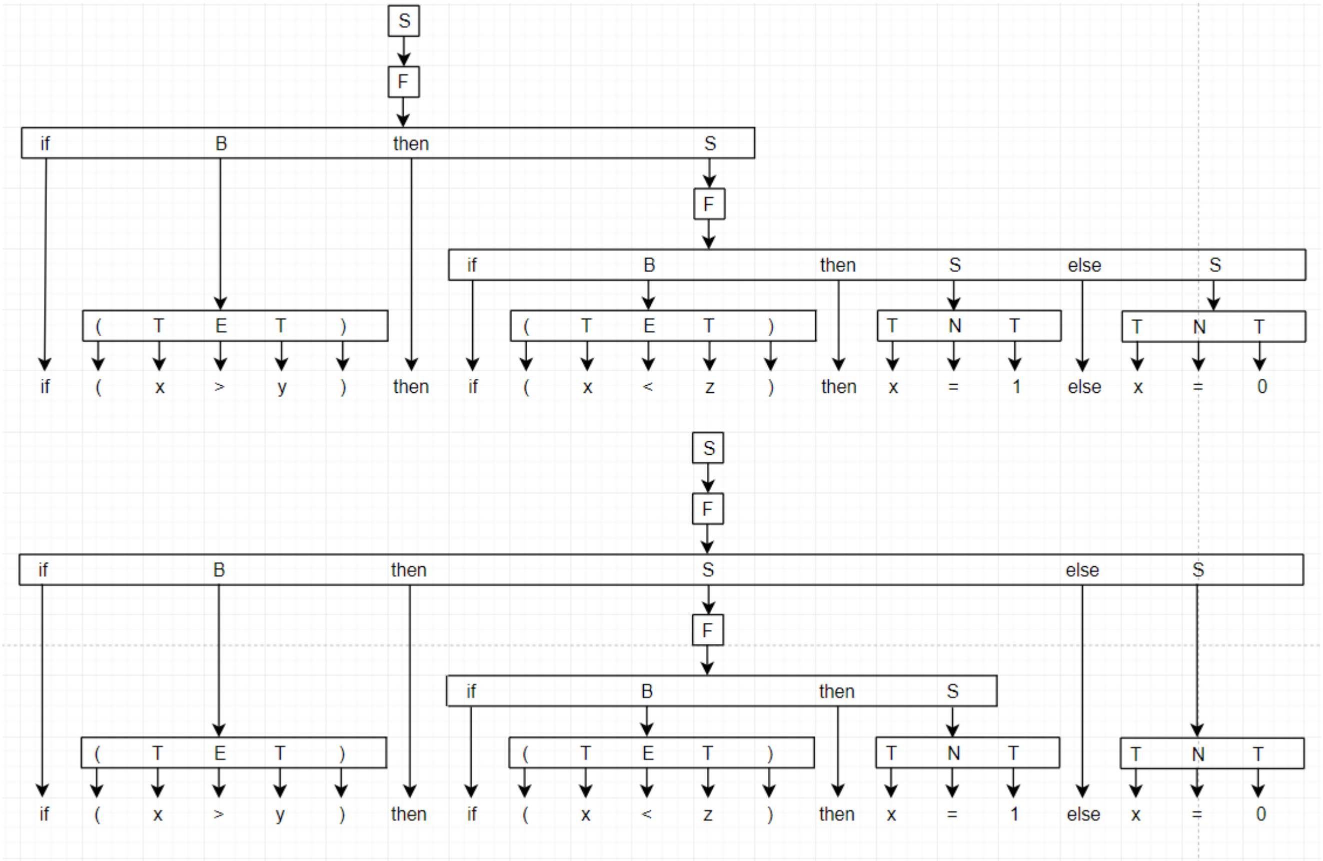
c) S ⇒ SA ⇒ Sc ⇒ SDc ⇒ S0c ⇒ SD0c ⇒ S10c ⇒ SA10c ⇒ Sb10c ⇒ SDb10c ⇒ S0b10c ⇒ A0b10c ⇒ a0b10c

d)



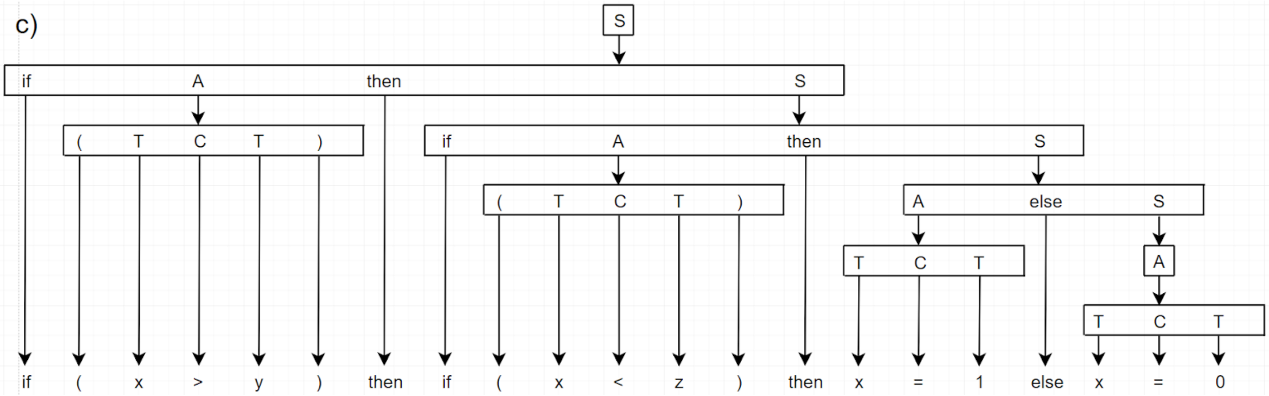
e) 1] “e” 2] “is” 3] ”odd”

2a)



b) **S** → if **A** then **S** | **A** else **S** | **A**
A → (**TCT**) | **TCT**
C → >, <, +, -, =
T → x, y, z, 0, 1

c)



d) These new grammar rules remove the ambiguity involved with F's "if B then S" and "if B then S else S". This allowed nested if/else statements to be represented through multiple avenues, as seen in my answer to 2a. I have removed the second option and replaced it with a smaller 'else' extension, as well as cleaning up other grammar rules.

3a) All four operators are left-to-right associative, as that is how they are defined in mathematics. For example, 1/2/3 ⇔ (1/2)/3 and not 1/(2/3) as the latter would be undefined with integer mathematics.

b) According to standard charts (one given here: https://ee.hawaii.edu/~tep/EE160/Book/chap5/subsection2.1.4.1.html),
The precedence of ‘*’ and ‘/’ are 12, and ‘+’ and ‘-’ are 11. This is parallel with the order of operations, where multiplication and division take precedence over addition and subtraction.

c) Here is a step by step walkthrough of exactly what is displayed by the parse tree.
Note: the ‘Rule’ column refers to the production rules, with rule 1.1 referring to E → E+T.

String	Explanation	Rule
“E”	The string begins as the starter symbol ‘E’	--
“E-T”	E_1 is turned into “E-T”	1.2
“T-T”	E_1 is turned into “T”	1.3
“F*T-T”	T_1 is turned into “F*T”	2.1
“y*T-T”	F_1 is turned into “y”	3.2
“y*F/T-T”	T_1 is turned into “F/T”	2.2
“y*x/T-T”	F_1 is turned into “x”	3.1
“y*x/F-T”	T_1 is turned into “F”	2.3
“y*x/z-T”	F_1 is turned into “z”	3.3
“y*x/z-F*T”	T_1 is turned into “F*T”	2.1
“y*x/z-z*T”	F_1 is turned into “z”	3.3
“y*x/z-z*F/T”	T_1 is turned into “F/T”	2.2
“y*x/z-z*z/T”	F_1 is turned into “z”	3.3
“y*x/z-z*z/F”	T_1 is turned into “F”	2.3
“y*x/z-z*z/y”	F_1 is turned into “y”	2.2

4a)

S → SS | TC | T
C → + | *
T → 0 | 1 | 2

b)

S → AC
A → 0
B → 1
C → AC | ACB | ϵ

c)

1 → 2
2 → 7 | 3
3 → 4 | 5
4 → 6
5 → 6
6 → 2