Programming Language Compilation
Midterm-2
CSC 340
KSU (2nd term 2017-2018)

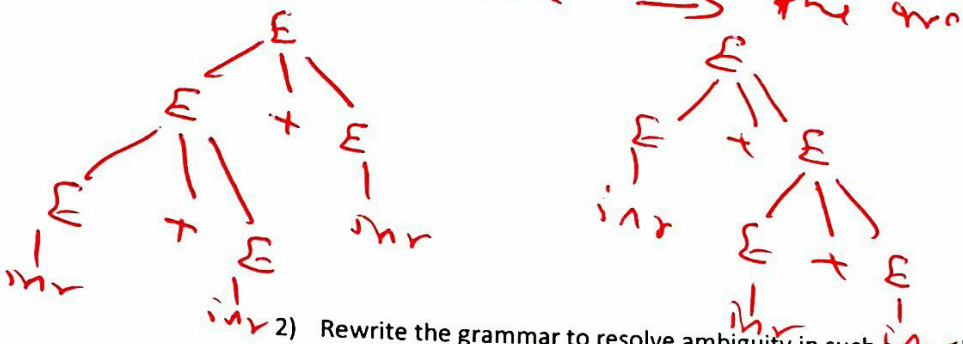
Student Name: **Student Number:**

Q1) A) Consider the following CFG

E→E+E|int

1) Show that the grammar is ambiguous. (2 marks)

the string int + int + int has more than one parse tree ⇒ the grammar is ambiguous



2) Rewrite the grammar to resolve ambiguity in such a way that + has left associativity. (3 marks)

E → E + int | int

B) Rewrite the following CFG so that we can write a recursive descent parser for.

E→E+E | E * E | id (2 marks)

* we need to eliminate Left recursion

E → id X | ε

X → + E X | * E X | ε

C) Rewrite the following CFG so that we can write an LL(1) for.

E→ id+E | id * E | (E) | id (2 marks)

* we need to left factor the grammar

E → id X | (E)

X → + E | * E | ε

D) Write a recursive descent parser in pseudo code (or c or Java code) for the grammar

E→ id+E | id * E | (E) | id (3 marks)

bool term(Token tok) { return *next++ == tok; }

E() { Token *save = next;
return (next=save, E1() ||
next=save, E2() ||
next=save, E3() ||
next=save, E4()) }

E1() { return term('id') && term('+') && E(); }

E2() - - -

E3() { }

E4() { }

Q2 A) Consider the following CFG

E→E+T|T
T→ T*F | F
F→ (E) | id

1. **Find the first and follow set of E** (2 marks)

First(E) = { (, 'id }

Follow(E) =

{ $, +,) }

2. **Find the first and follow set of F** (2 marks)

First(F) = { (, 'id }

Follow(F) =

{ $, +,), * }

{ (, $ }

3. **Find the first and follow set of ')'** (2 marks)

First(')') :

= {) }

Follow(')') = follow(F) = follow(T) ∪ follow(E)

{ $, +,), * }

B) Assume that a grammar contains the production E→T+F|TF. Assume also that first(T)={id, ε}, fist(F)= { * , ε }, follow(E)={id,$} and follow(T)={+, $}

1. Place the productions **E→T+F and E→TF in the right entries of the LL(1) table.** (3 marks)

	id	*	+	$
E	T+F / TF	TF	T+F	TF

2. **Is the table LL(1)? Why? Or Why not?** (2 marks)

Not LL(1) because entry [E, id] is multiply defined

Q3. **Assuming that there is a state in the DFA for recognizing the viable prefixes of a grammar contains the items.**

E→T.+F
T→F.

a) Under what condition would the SLR algorithm reduce? (2 marks)

if the next input token is in the follow(

b) Under what condition would the SLR algorithm shift? (2 marks)

If the next input token is '+'

c) Assuming the follow(T)=follow(F)={+,*,$}, is the grammar SLR? Why or why not? (2 marks)

No because + ~~is~~ '+' ∈ follow(T)

Q4) Draw the symbol table when line 14 in the following code is processed. (3 marks)

```
 0: int x = 137;
 1: int z = 42;
 2: int MyFunction(int x, int y) {
 3:   printf("%d,%d,%d\n", x, y, z);
 4:   {
 5:     int x, z;
 6:     z = y;
 7:     x = z;
 8:     {
 9:       int y = x;
10:       {
11:         printf("%d,%d,%d\n", x, y, z);
12:       }
13:       printf("%d,%d,%d\n", x, y, z);
14:     }
15:     printf("%d,%d,%d\n", x, y, z);
16:   }
17: }
```