Each Question 12 points
Exam out of 72 points

KEY

جامعة بيرزيت
BIRZEIT UNIVERSITY

College of Engineering & Technology
Computer Science Department

(12)

Comp439 **Midterm Exam** **22/11/2018**

4 **[1] (a)** What are the programming languages views? Explain.

1- Designer (Inventor of the PL)

2- Implementor (Who develops & writes the compiler)

3- User (who writes programs in this PL)

4 **(b)** Explain Briefly the meaning and give examples of **Orthogonality** in a PL.

The PL should behave same thing in similar contexts.

EX: In IBM machines, Addition is performed:

A register, memory
AR register1, register2 > Less orthogonal

But in VAX machines, there is only one instruction.

ADDL operand1, operand2 > more orthogonal better

4 **(c)** What are the programming languages paradigms. Give example on each.

1. Imperative (procedural) paradigm : Pascal, C
2. Functional Paradigm: Lisp
3- Logical Paradigm: Prolog
4. Object Oriented Paradigm: Java

12

[2] (a) Write a function in Clisp, **min**(x y) which computes the minimum value of x and y.

```
> (defun min (x y)
      (if (> x y) y x))
```

6

(b) What is the output of the following function in Clisp, justify your answer by **tracing** the function call (func 17 3).

```
> (defun func (m n)
    ( if (> n m) 0
      (+ 1 (func (- m n) n ))))

> (func 17 3) = 5
```

6

```
(1 + (func 14 3)) = 5
   (1 + (func 11 3)) = 4
      (1 + (func 8 3)) = 3
         (1 + (func 5 3)) = 2
            (1 + (func 2 3)) = 1
                     0
```

what the function **func** do?

2 function func(m,n) computes & returns the value of m div n (m/n)

(12) **[3]** Given the following simple grammar:

block $\rightarrow$ **begin** decls stmts **end**
decls $\rightarrow$ **var** dec-item | λ
dec-item $\rightarrow$ **D** | **D ,** dec-item
stmts $\rightarrow$ statement **;** stmts | λ
statement $\rightarrow$ **S** | λ

V_T ={**begin, end, var, D, S,** ; , , } V_N ={block, decls, decl-item, stmts, statement }

(a) Give a program generated by this grammar.

4

begin
var D, D
S;
S;
S;
end

(b) Rewrite production rules using **EBNF** notations.

4

block → begin decls stmts end
decls → var dec-item | λ
dec-item → D{, D}*
stmts → (statement ;)*
statement → S | λ

(c) Compute **FOLLOW**(dec-item).

4

FOLLOW(dec-item) = FOLLOW(decls)
= FIRST(stmts) ∪ {end}
= {S, ;} ∪ {end} = {S, ;, end}

12

[4] (a) Given the following in C code:

```
int power(int m, int n);      // The function power computes and returns m^n

void main()
{  const int max=10;
   float x=1, y=10;
   x  += y;
   int p = power (max,2);
```

Draw the **symbol table** after the above code is executed.

8

Name	Type	Value
Power	function Name	100
max	integer Constant	10
x	float Variable	~~1~~ 11
y	float Variable	10
p	integer Variable	100

(b) Given the grammar:

$G \rightarrow S\$$
$S \rightarrow AS$
$A \rightarrow AAB \mid a \mid \lambda$
$B \rightarrow bBS \mid c \mid \lambda$

6

	FIRST	FOLLOW
G	a b c $	—
S	a b c	$ a b c
A	a b c λ	a b c
B	b c λ	a b c

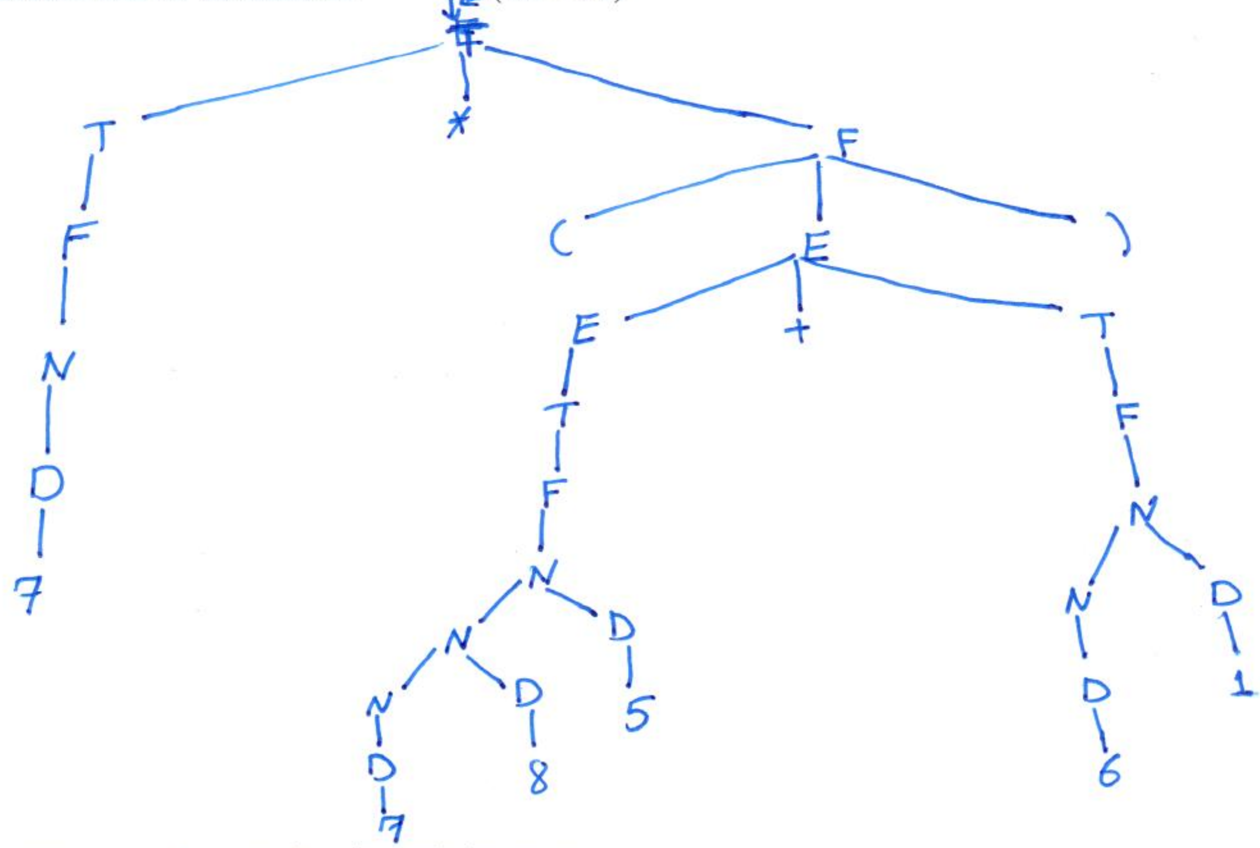
[5] **(a)** Given the grammar:

5

$$
\begin{aligned}
E &\longrightarrow E + T \;|\; T \\
T &\longrightarrow T * F \;|\; F \\
F &\longrightarrow (E) \;|\; N \\
N &\longrightarrow ND \\
N &\longrightarrow D \\
D &\longrightarrow 0 \,|\, 1 \,|\, 2 \,|\, \ldots \,|\, 9
\end{aligned}
$$

Draw the derivation tree for the sentence: **7 * (785 + 61)**

6



(b) Given the following DFSA. Reduce it to **minimum** states.

6

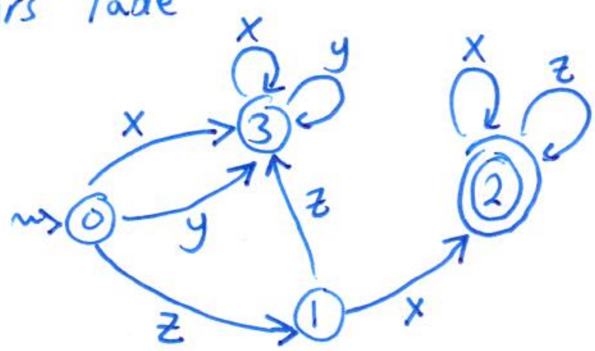
	x	y	z
(3,4)	(3,4)	(4,3)	
(2,5)	(6,6)		(5,6)
(2,6)	(6,2)		(5,5)
(5,6)	(6,2)		(6,5)

Feasible Pairs Table

δ	x	y	z
0	3	3	1
1	5		4
2	6		5
3	3	4	
4	4	3	
5	6		6
6	2		5

$3 \equiv 4$

$2 \equiv 5 \equiv 6$

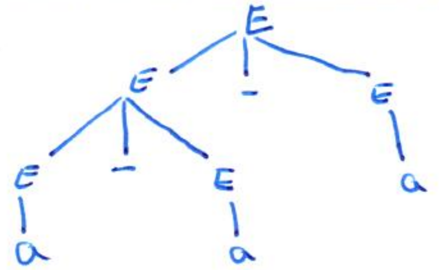


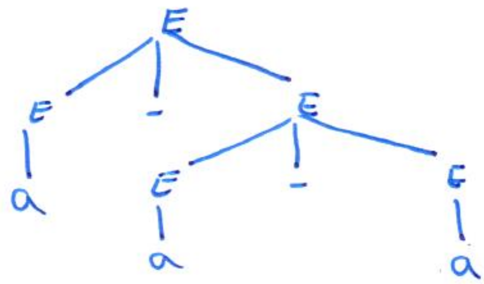
	x	y	z
0	3	3	1
1	2		3
2	2		2
3	3	3	

12 **[6] Given the grammar:**

E → E – E | (E) | a

(a) Show that the grammar is **ambiguous**.





a-a-a

a-a-a

6

Two derivation trees

(b) A student transform the above grammar to the following none ambiguous grammar:

E → T – E | T

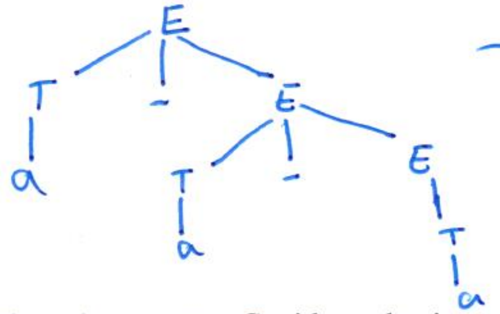
T → (E) | a

What is wrong with code. Explain.

This grammar is right associative which contradicts the associative rules in the "–" operations

consider the sentence a-a-a , its derivation tree is:

6



This mean that a-a-a will be executed as a-(a-a) which contradict the associativity rule.

(c) Given the grammar G with productions:

A → α

A → β

A → λ , where $\alpha, \beta \neq \lambda$

State explicitly the conditions so that the above grammar is LL(1).

ملغي