

$\text{Addr} \rightarrow \text{Name} @ \text{id} . \text{Name}$

$\text{Name} \rightarrow \text{id Name}'$

$\text{Name}' \rightarrow \epsilon \mid . \text{id Name}'$

$\text{id} \rightarrow \text{"Mahdieh"} \mid \text{"98522076"}$

مرحله ۱) چک کردن left factoring:

در ۳ non terminal موجود بررسی میکنیم. تنها سطر آخر (|) بین دو قانون وجود دارد که چون یکی  $\epsilon$  است اصلا فاکتور گیری منطقی و ممکن نیست.

مرحله ۲) چک کردن left recursion:

در اینجا چون هیچ قانونی با فرمت  $A \rightarrow Aa \mid b$  وجود ندارد پس همچین مشکلی نداریم.

مرحله ۳) چک کردن Nullable:

فقط در سطر آخر گرامر  $\epsilon$  وجود دارد پس مجموعه ی First و Follow برای  $\text{Name}'$  را پیدا میکنیم:

$\text{First}(\text{Name}') = \text{First}(\epsilon \mid . \text{id Name}') = \{., \epsilon\}$

$\text{Follow}(\text{Name}') = \text{Follow}(\text{Name}) = \{ @, \$ \}$

اشتراک مجموعه ها را حساب میکنیم و برابر با تهی میشود چون هیچ عضو مشترکی ندارند.

در نتیجه اثبات میشود گرامر LL1 است.

Parse Table:

First و Follow برای همه ی Non Terminal ها را بدست می آوریم:

$\text{First}(\text{Name}) = \text{First}(\text{id Name}') = \text{First}(\text{id}) = \{ \text{Mahdieh}, 98522076 \}$

$\text{Follow}(\text{Name}) = \{ @, \$ \}$

$\text{First}(\text{Addr}) = \text{First}(\text{Name} @ \text{id} . \text{Name}) = \text{First}(\text{Name}) = \{ \text{Mahdieh}, 98522076 \}$

$\text{Follow}(\text{Addr}) = \{ \$ \}$

		@	.	Mahdieh	98522076	\$
Addr				Name @ id . Name	Name @ id . Name	
Name				id Name'	id Name'	
Name'		$\epsilon$	. id Name'	Name' $\rightarrow \epsilon$	Name' $\rightarrow \epsilon$	
id				Mahdieh	98522076	

Id = Mahdieh

$\text{Name}' \rightarrow \epsilon \mid . \text{id Name}' \rightarrow \text{Name}' = .\text{Mahdieh}$

Name → id Name' → Name = Mahdieh.Mahdieh

Addr → Name @ id . Name → Addr = Mahdieh.Mahdieh@Mahdieh.Mahdieh.Mahdieh

Input = Mahdieh@98522076.Mahdieh

number	stack	input	prod
1	Addr	<u>Mahdieh@98522076.Mahdieh</u> \$	Name @ id . Name
2	Name.id@Name	<u>Mahdieh@98522076.Mahdieh</u> \$	Id Name'
3	Name.id@ Name'id	<u>Mahdieh@98522076.Mahdieh</u> \$	Mahdieh
4	<u>Name.id@</u> Name' Mahdieh	<u>Mahdieh@98522076.Mahdieh</u> \$	Del
5	Name.id@Name'	<u>@98522076.Mahdieh</u> \$	ε
6	Name.id@	<u>@98522076.Mahdieh</u> \$	Del
7	Name.id	<u>98522076.Mahdieh</u> \$	98522076
8	Name.98522076	<u>98522076.Mahdieh</u> \$	Del
9	Name.	<u>.Mahdieh</u> \$	Del
10	Name	<u>Mahdieh</u> \$	id Name'
11	Name'id	<u>Mahdieh</u> \$	Mahdieh
12	Name'Mahdieh	<u>Mahdieh</u> \$	Del
13	Name'	<u>ε</u> \$	Del
14		<u>\$</u>	

سوال دوم:

در پوشه ی Midterm در پوشه ی grammar در فایل email.g4

The screenshot shows an IDE with the following components:

- Project Explorer:** Shows a project named 'Midterm' with a folder 'gen' containing files like 'emailLexer.interp', 'emailLexer.tokens', 'emailParser.py', etc., and a folder 'grammar' containing 'email.g4'.
- Editor:** Displays the content of 'email.g4':

```
grammar email;
start: EMAIL EOF;

EMAIL: LOCAL_SUBPART ('.' LOCAL_SUBPART)* '@' DOMAIN_SUBPART ('.' DOMAIN_SUBPART)*;

fragment LOCAL_SUBPART : [a-zA-Z0-9!$%()*+,-;:_~]*;
fragment DOMAIN_SUBPART : [a-zA-Z0-9-]*;
```
- ANTLR Preview:** Shows the input 'm\_nader198@comp.iut.ac.ir' and the generated parse tree:

```
graph TD
    start --> EMAIL[m_nader198@comp.iut.ac.ir]
    start --> EOF[EOF]
```
- Console:** Displays the message 'parser for email.g4 generated to E:/01021/Compiler/Midterm/gen'.