CS224 Final 472

Part 1: [8 marks]

Consider the alphabet {a-z, 0-9} and the following regular definitions;

```
FNUM → [0–9]² (.[0–9])?
VAR  → [a–c] [0–2]
```

Q1- Draw the Finite Automaton for each of the two regular definition. [6 marks]

Q2- Give one shortest string from each regular expression. [2 marks]

FNUM: **VAR:**

Part 2: [42 marks]

Use the following grammar to answer the questions [Q3 to Q6]

```
stmt → while (expr) : stmt
     | while expr : stmt
     | VAR = expr
expr → expr * expr | VAR | FNUM
```

Give the equivalent grammar after solving each of the following questions. If you add new rules name them starting from X, Y, Z, T ...

Q3- Use the sentence below to prove that the grammar is ambiguous. [6 marks]

```
VAR = FNUM * VAR * FNUM
```

Q4- Remove ambiguity fromt the grammar (only include the parts you rewrite) **[2 marks]**

Q5- Remove left recursion (only include the parts you rewrite) **[4 marks]**

Q6- Remove left factoring (only include the parts you rewrite) **[8 marks]**

Use the following grammar to answer the questions [Q7 & Q8]

```
stmt → while (expr) : stmt
     | for VAR in range (FNUM) : stmt
     | VAR = expr
expr → VAR | FNUM
```

Q7- Write the First and Follow sets for the following non-terminals: **[8 marks]**

First(stmt)		Follow(stmt)	
First(expr)		Follow(expr)	

Q8- Write the following functions: **[14 marks]**

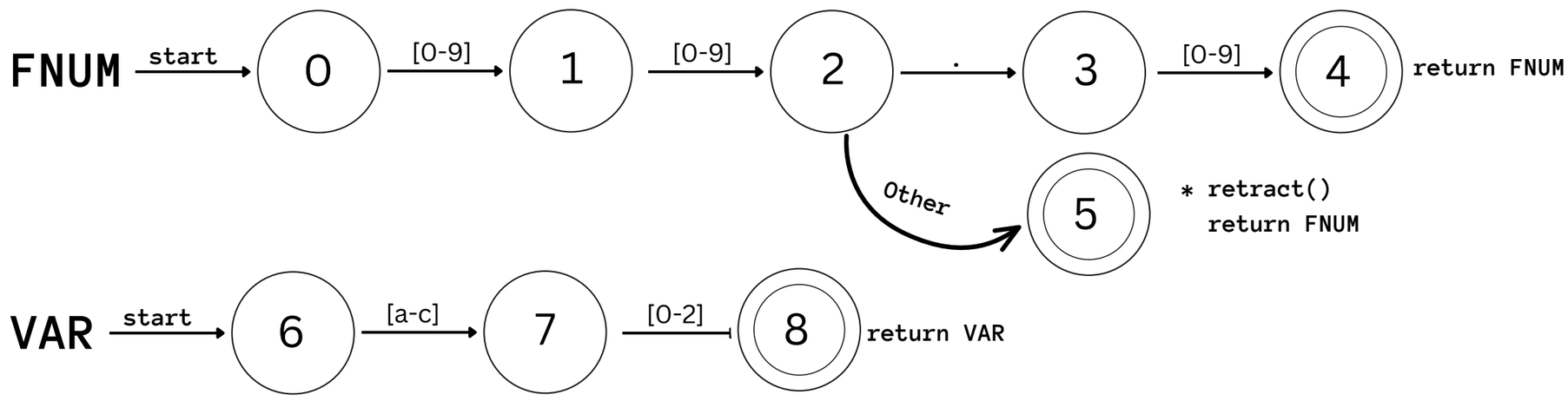
def stmt():	def expr():

Answers are on the next page

Answers

Question 1 answer:

ملاحظة: هذا ليس الحل الوحيد الصحيح، وقد يكون الحل المعتمد لدى المنسق مختلف بعض الشيء.

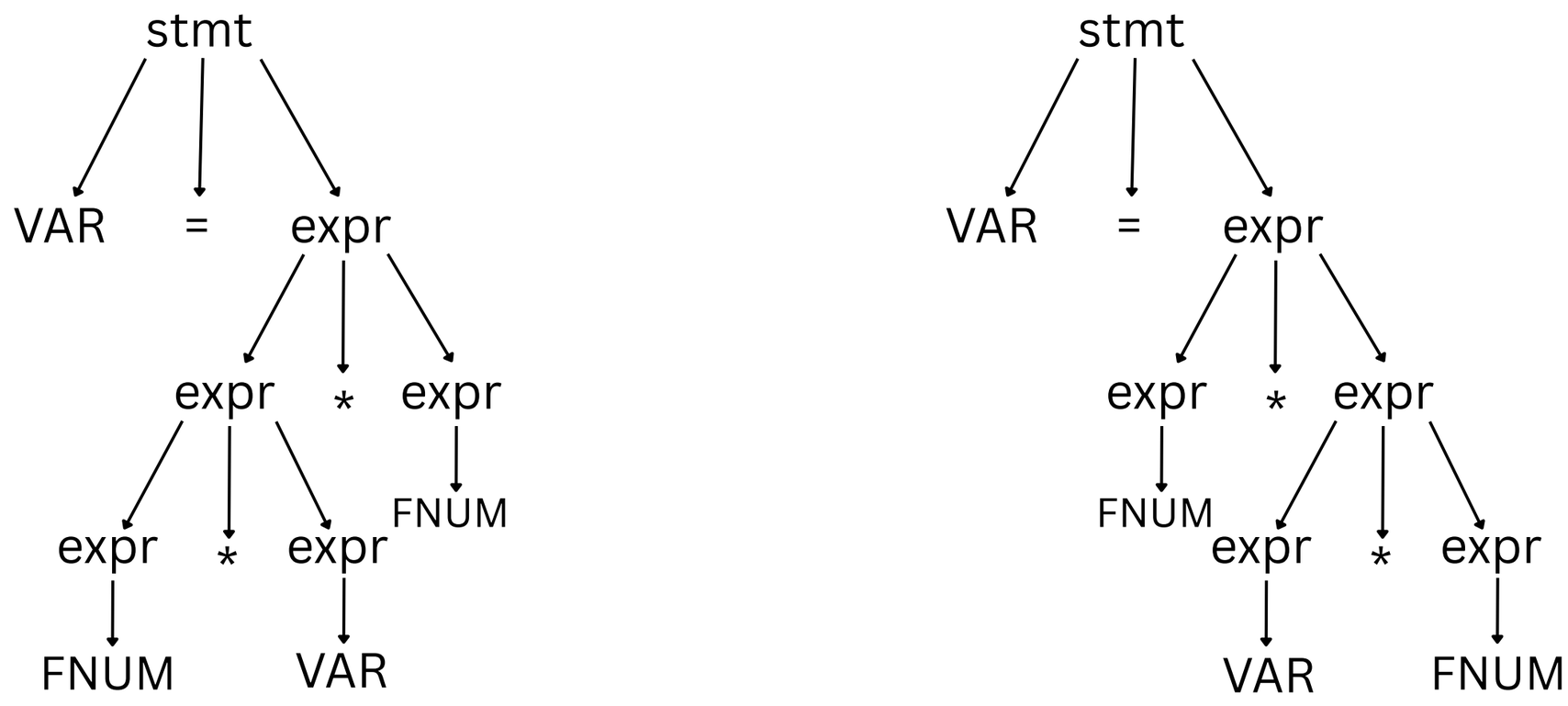


Question 2 answer:

`FNUM:67`

`VAR:b1`

Question 3 answer:



Question 4 answer:

```
expr → expr * X
     | X
X    → VAR | FNUM
```

Question 5 answer:

```
expr → VAR Y | FNUM Y
Y    → * expr Y | ε
```

Question 6 answer:

```
stmt → while Z
     | VAR = expr
Z    → (expr) : stmt
     | expr : stmt
```

Question 7 answer:

First(stmt)	while , for , VAR	Follow(stmt)	$
First(expr)	VAR , FNUM	Follow(expr)	) , $

Question 8 answer:

```
stmt → while (expr) : stmt
      | for VAR in range (FNUM) : stmt
      | VAR = expr
expr → VAR | FNUM
```

```
def stmt():
 match lookahead:
    case "while":
        match("while")
        match("(")
        expr()
        match(")")
        match(":")
        stmt()
    case "for":
        match("for")
        match("VAR")
        match("in")
        match("range")
        match("(")
        match("FNUM")
        match(")")
        match(":")
        stmt()
    case "VAR":
        match("VAR")
        match("=")
        expr()
    case _:
        error(“syntax error”)
```

```
def expr():
 match lookahead:
     case "VAR":
         match("VAR")

     case "FNUM":
         match("FNUM")

     case _:
         error(“syntax error”)
```