Date

THEORY OF AUTOMATA (Computer Science)

BSCS - 501

Semester Project

[illegible] 26/Apr

1) Ahmed Ali (B20102013)

2) Muhammad Amas (B20102077)

3) Mariya Suleman (B20102065)

4) Urooba Shameem (B20102180)

5) Zunain Ali Azam (B20102183)

L.Sp = 9
CFU = 10

Submitted to:
Ms. Madiha Khurram

Date

RULES FOR NAMING A VARIABLE:-

A variable name is a word that consist of following:

English letter, A-Z & a-z.

Digits, 0-9

An underscore character (_).

A dollar sign ($)

No other character is permitted.

Space Not permitted.

Variable must not start with digits or underscore (0 / _). only. A character is necessary after that.

Double underscores are not allowed.

Example:-

.1a.	X
$my_	X
num $	X
12	✓
1num	✓

Language Specification

LANG X

Introduction :-

To define the language we have to give the details about the following:

→ Data types
→ Operators
→ Keywords
→ Punctuation
→ Conditional statement
→ Loops
→ Functions

Data types :-

• Integer	→ num
• Floating point	→ decimal
• Character	→ letter

Punctuation :-

→ ( )
→ [ ]
→ ,
→ ' '

Conditional Statement

→ either-or
→ option

Loops :-
→ for
→ jump

Function :-
→ define function-name (parameters)

Operators :-
→ Arithmetic (+, -, /, *)
→ Increment (++) → Decrement (--)
→ Comparision (<, >, >=, <=, ==, !=)
→ Assignment (=)

Input & Output :-
→ Input : take( )
→ Output : show( )

Keywords :-

num	take
decimal	show
letter	
either	
or	
jump	
for	
option	
change	
default	
return	
label	
define	

Token	Class Part
(	(
)	)
=	=
+	+
-	-
÷	/
x	*
{}	[]
;	;
>, <, >=, <=, ==	>, <, >=, <=, ==
!=	!=

Date

Variable Initilization & Declaration :-

< Data type > <variable name > <assignment operator> <value >

example :

```
num value1 = 30
letter abc
```

Conditional Statements :-

either -or

```
either (<condition >)
 [
     <body>
 ]
or
 [
     <body>
 ]
```

example :-

```
either (a==4)
[
  Show ('four')
 ]
or
[
  show ('It is not four')
]
```

option

```
option (<parameters)
   change (<value>)
    [ <body> ]
   change (<value>)
    [ <body> ]
    ;
   default
     [ <body> ]
```

example :-

```
option (~~int~~ a)
 change (1)
  [ show('one') ]
 change (2)
  [show ('two') ]
 change (3)
  [ Show ('three') ]
 default
  [ Show ('other than
      one two three') ]
```

loops:-

for

```
for (<initilize>, <conditional statement>,
     <increment / decrement>)
 [
    <body>
 ]
```

example :-

```
for (num val = 0, val < 10, val++)
[
   show (val)
]
```

jump

```
label <name>
     <body>
jump <name>
    <body>
```

example :-

```
label L1
   show ('z')
jump L1
```

Function:-

```
define <function-name>(<parameters>)
 [
    <body>
    return (<statement>)
 ]
```

example:-

```
define add(decimal a , decimal b)
[
    decimal c = decimal a + decimal b
    return(c)
]
```

Date

Declaration Of Variables:-

<decl> → <DT> ID < New> <assignment operator>

~~<DT> → int | float | char~~

<DT> → num | decimal | lettel

< values → -? [0-9]+ (|. [0-9]+) ? |

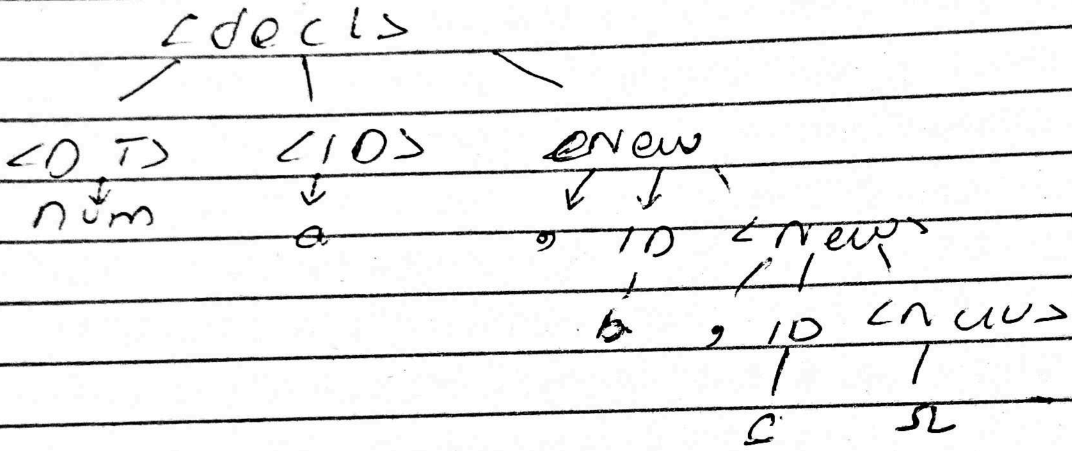
[a-z] | Ω → i.e

⇒ lettel

ID = (- | L)

<New> → , ID < New> | Ω ⇒ Ω = null

Parse tree :-



Delivation:-

< decl > → <DT> ID < New>

num a , ID < New>

num a, b, ID <New>

num a, b, c

Functions:-

define <functionname> (<parametees>)

Delivation:-

(<parametees>) → <DT> ID < New> | Ω

< New> → <DT> ID < New > | Ω

Date

Eithel - or :-

~~<datatype> ID <ass~~

<cond> -> <DT> ID <cond-op> <values

<DT> num | decimal | lettee | Ω

<cond -op> -> < | > | == | <= | !=

Date

: For Loop:

<for-loop> → for (<initialize>, <conditional statement>, <inc/dec>)

<initialize> → <DT> ID = val

<conditional statement> → ID <conditional-operator> <val> ~~<inc/de~~

<inc/dec> → ID <op> / <op> ID

<op> → ++ / --

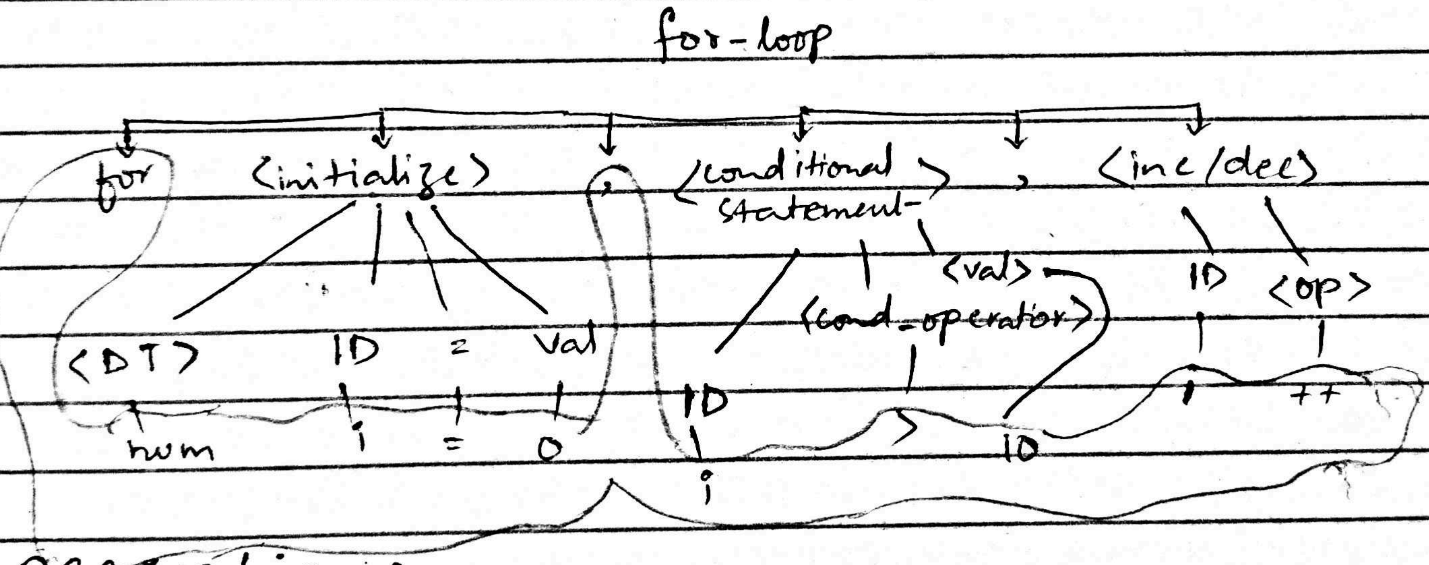
<conditional_operator> → < / > / <= / >= / == / !=

<val> → ID / num / decimal

<DT> → num / decimal

ID → (-/l)

: Parse Tree:



Derivation:-

<for loop> → for (<init>, <cond>, <inc/dec>)

→ for (<DT> ID = val ; ID <cond-op> <val>, ID <op>)

→ for (num i = 0 ; i < 10}, i++)