

# Bài 4

# BNF và sơ đồ cú pháp

ONE LOVE. ONE FUTURE.

# Công thức siêu ngữ Backus

- **Siêu ngữ (metalinguage )**: Ngôn ngữ sử dụng các lệnh để mô tả ngôn ngữ khác
- **BNF (Backus Naur Form)** là dạng siêu cú pháp để mô tả các ngôn ngữ lập trình
- BNF được sử dụng rộng rãi để mô tả văn phạm của các ngôn ngữ lập trình, tập lệnh và các giao thức truyền thông.

# Ký pháp BNF

- Ký pháp BNF là một tập các luật ,vết trái của mỗi luật là một cấu trúc cú pháp.
- Tên của cấu trúc cú pháp được gọi là ký hiệu không kết thúc.
- Các ký hiệu không kết thúc thường được bao trong cặp <>.
- Các ký hiệu kết thúc thường được phân cách bằng cặp nháy đơn hoặc nháy kép



# Ký pháp BNF

- Mỗi ký hiệu không kết thúc được định nghĩa bằng một hay nhiều luật.
- Các luật có dạng

$N ::= s$

(*N là ký hiệu không kết thúc, s là một xâu gồm 0 hay nhiều ký hiệu kết thúc và không kết thúc. Các luật có chung về trái được phân cách bằng |*)



## Ví dụ về BNF : văn phạm sản sinh các số thực

```
<đơn vị> ::= <khoảng cách><số tự nhiên> |  
          <khoảng cách><số tự nhiên>'.'<dãy chữ số> |  
          <khoảng cách>'.'<chữ số>< dãy chữ số > |  
          <số thực>'e'<số tự nhiên>  
<khoảng cách> ::= ε | '+' | '-'  
<số tự nhiên> ::= '0' | <chữ số khác 0><dãy chữ số>  
<chữ số khác 0> ::= '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'  
< dãy chữ số > ::= ε | <chữ số>< dãy chữ số >  
<chữ số> ::= '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
```

Văn phạm này được viết bằng BNF, một công cụ rất phổ biến để biểu diễn cú pháp ngôn ngữ lập trình

- EBNF (Extended BNF ) được phát triển từ ký pháp BNF. EBNF có ký pháp tương tự BNF nhưng được đơn giản hóa bằng cách sử dụng một số ký hiệu đặc biệt :

[] phần này là tuỳ chọn(có hoặc không)

{ } phần này có thể lặp lại một số lần tuỳ ý hoặc không xuất hiện lần nào (Nếu lặp lại m hay n lần , dùng n hay m là chỉ số trên hoặc dưới)

Không cần dùng “ cho ký hiệu kết thúc

# So sánh BNF và EBNF

## Ví dụ

- Trong EBNF

$\langle \text{Lệnh if} \rangle ::= \text{IF } \langle \text{Biểu thức} \rangle \text{ THEN } \langle \text{Lệnh} \rangle [\text{ELSE } \langle \text{Lệnh} \rangle]$

- Trong BNF

$\langle \text{Lệnh if} \rangle ::= \text{'IF' } \langle \text{Biểu thức} \rangle \text{ 'THEN' } \langle \text{Lệnh} \rangle \mid \text{'IF' } \langle \text{Biểu thức} \rangle \text{ 'THEN' } \langle \text{Lệnh} \rangle \text{ 'ELSE' } \langle \text{Lệnh} \rangle$



## Ví dụ: Một đoạn văn phạm Python trên EBNF

```
compound_stmt: if_stmt | while_stmt | for_stmt | try_stmt | with_stmt  
| funcdef | classdef | decorated | async_stmt  
async_stmt: 'async' (funcdef | with_stmt | for_stmt)  
if_stmt: 'if' test ':' suite ('elif' test ':' suite)* ['else' ':' suite]  
while_stmt: 'while' test ':' suite ['else' ':' suite]  
for_stmt: 'for' exprlist 'in' testlist ':' suite ['else' ':' suite]  
try_stmt: ('try' ':' suite  
          ((except_clause ':' suite)+  
           ['else' ':' suite]  
           ['finally' ':' suite] |  
            'finally' ':' suite))  
with_stmt: 'with' with_item ("," with_item)* ':' suite  
with_item: test ['as' expr]
```

# Văn phạm KPL viết bằng BNF

```
01) <Prog> ::= KW_PROGRAM TK_IDENT SB_SEMICOLON <Block> SB_PERIOD  
02) <Block> ::= KW_CONST <ConstDecl> <ConstDecls> <Block2>  
03) <Block> ::= <Block2>  
  
04) <Block2> ::= KW_TYPE <TypeDecl> <TypeDecls> <Block3>  
05) <Block2> ::= <Block3>  
  
06) <Block3> ::= KW_VAR <VarDecl> <VarDecls><Block4>  
07) <Block3> ::= <Block4>  
  
08) <Block4> ::= <SubDecls><Block5>|<Block5>  
09) <Block5> ::= KW_BEGIN <Statements> KW_END  
  
10) <ConstDecls> ::= <ConstDecl> <ConstDecls>  
11) <ConstDecls> ::= ε  
12) <ConstDecl> ::= TK_IDENT SB_EQUAL <Constant> SB_SEMICOLON  
  
13) <TypeDecls> ::= <TypeDecl> <TypeDecls>  
14) <TypeDecls> ::= ε  
15) <TypeDecl> ::= TK_IDENT SB_EQUAL <Type> SB_SEMICOLON  
  
16) <VarDecls> ::= <VarDecl> <VarDecls>  
17) <VarDecls> ::= ε  
18) <VarDecl> ::= TK_IDENT SB_COLON <Type> SB_SEMICOLON
```

# Văn phạm KPL viết bằng BNF

```
19) <SubDecls> ::= <FunDecl> <SubDecls>
20) <SubDecls> ::= <ProcDecl> <SubDecls>
21) <SubDecls> ::= ε

22) <FunDecl> ::= KW_FUNCTION TK_IDENT <Params> SB_COLON <BasicType>
   SB_SEMICOLON <Block> SB_SEMICOLON

23) <ProcDecl> ::= KW PROCEDURE TK_IDENT <Params> SB_SEMICOLON <Block>
   SB_SEMICOLON

24) <Params> ::= SB_LPAR <Param> <Params2> SB_RPAR
25) <Params> ::= ε

26) <Params2> ::= SB_SEMICOLON <Param> <Params2>
27) <Params2> ::= ε

28) <Param> ::= TK_IDENT SB_COLON <BasicType>
29) <Param> ::= KW_VAR TK_IDENT SB_COLON <BasicType>

30) <Type> ::= KW_INTEGER
31) <Type> ::= KW_CHAR
32) <Type> ::= TK_IDENT
33) <Type> ::= KW_ARRAY SB_LSEL TK_NUMBER SB_RSEL KW_OF <Type>
```



# Văn phạm KPL viết bằng BNF

```
34) <BasicType> ::= KW_INTEGER  
35) <BasicType> ::= KW_CHAR  
  
36) <UnsignedConstant> ::= TK_NUMBER  
37) <UnsignedConstant> ::= TK_IDENT  
38) <UnsignedConstant> ::= TK_CHAR  
  
40) <Constant> ::= SB_PLUS <Constant2>  
41) <Constant> ::= SB_MINUS <Constant2>  
42) <Constant> ::= <Constant2>  
43) <Constant> ::= TK_CHAR  
  
44) <Constant2> ::= TK_IDENT  
45) <Constant2> ::= TK_NUMBER  
  
46) <Statements> ::= <Statement> <Statements2>  
  
47) <Statements2> ::= SB_SEMICOLON <Statement> <Statements2>  
48) <Statements2> ::= ε
```



# Văn phạm KPL viết bằng BNF

```
49) <Statement> ::= <AssignSt>
50) <Statement> ::= <CallSt>
51) <Statement> ::= <GroupSt>
52) <Statement> ::= <IfSt>
53) <Statement> ::= <WhileSt>
54) <Statement> ::= <ForSt>
55) <Statement> ::= ε
56) <AssignSt> ::= <Variable> SB_ASSIGN <Expression>
57) <AssignSt> ::= TK_IDENT SB_ASSIGN <Expression>

58) <CallSt> ::= KW_CALL TK_IDENT <Arguments>

59) <GroupSt> ::= KW_BEGIN <Statements> KW_END

60) <IfSt> ::= KW_IF <Condition> KW_THEN <Statement> <ElseSt>
61) <ElseSt> ::= KW_ELSE <Statement>
62) <ElseSt> ::= ε

63) <WhileSt> ::= KW WHILE <Condition> KW DO <Statement>
64) <ForSt> ::= KW FOR TK_IDENT SB_ASSIGN <Expression> KW_TO
                <Expression> KW DO <Statement>
```



# Văn phạm KPL viết bằng BNF

```
65) <Arguments> ::= SB_LPAR <Expression> <Arguments2> SB_RPAR
66) <Arguments> ::= ε

67) <Arguments2> ::= SB_COMMA <Expression> <Arguments2>
68) <Arguments2> ::= ε

68) <Condition> ::= <Expression> <Condition2>

69) <Condition2> ::= SB_EQ <Expression>
70) <Condition2> ::= SB_NEQ <Expression>
71) <Condition2> ::= SB_LE <Expression>
72) <Condition2> ::= SB_LT <Expression>
73) <Condition2> ::= SB_GE <Expression>
74) <Condition2> ::= SB_GT <Expression>
```



# Văn phạm KPL viết bằng BNF

```
75) <Expression> ::= SB_PLUS <Expression2>
76) <Expression> ::= SB_MINUS <Expression2>
77) <Expression> ::= <Expression2>

78) <Expression2> ::= <Term> <Expression3>

79) <Expression3> ::= SB_PLUS <Term> <Expression3>
80) <Expression3> ::= SB_MINUS <Term> <Expression3>
81) <Expression3> ::= ε

82) <Term> ::= <Factor> <Term2>

83) <Term2> ::= SB_TIMES <Factor> <Term2>
84) <Term2> ::= SB_SLASH <Factor> <Term2>
85) <Term2> ::= ε
86) <Factor> ::= <UnsignedConstant>
87) <Factor> ::= <Variable>
88) <Factor> ::= <FunctionApptication>
89) <Factor> ::= SB_LPAR <Expression> SB_RPAR

90) <Variable> ::= TK_IDENT <Indexes>
91) <FunctionApplication> ::= TK_IDENT <Arguments>

92) <Indexes> ::= SB_LSEL <Expression> SB_RSEL <Indexes>
93) <Indexes> ::= ε
```



- Đọc vào: các hàm

- ReadI: Đọc 1 số nguyên. Không tham số
- ReadC: Đọc 1 ký tự. Không tham số

Ví dụ

```
var a: integer;  
a:= ReadI;
```

- In ra: các thủ tục

- Writeln: In ra 1 số nguyên. 1 tham số
- WriteC: In ra 1 ký tự. 1 tham số
- WriteLn: In dấu xuống dòng. Không tham số

Ví dụ

```
call Writeln(a);  
call WriteLn;
```

# Hàm tính tổng hai số nguyên

```
function sum (x: integer; y:integer): integer;  
begin  
    sum:=x+y;  
end;
```

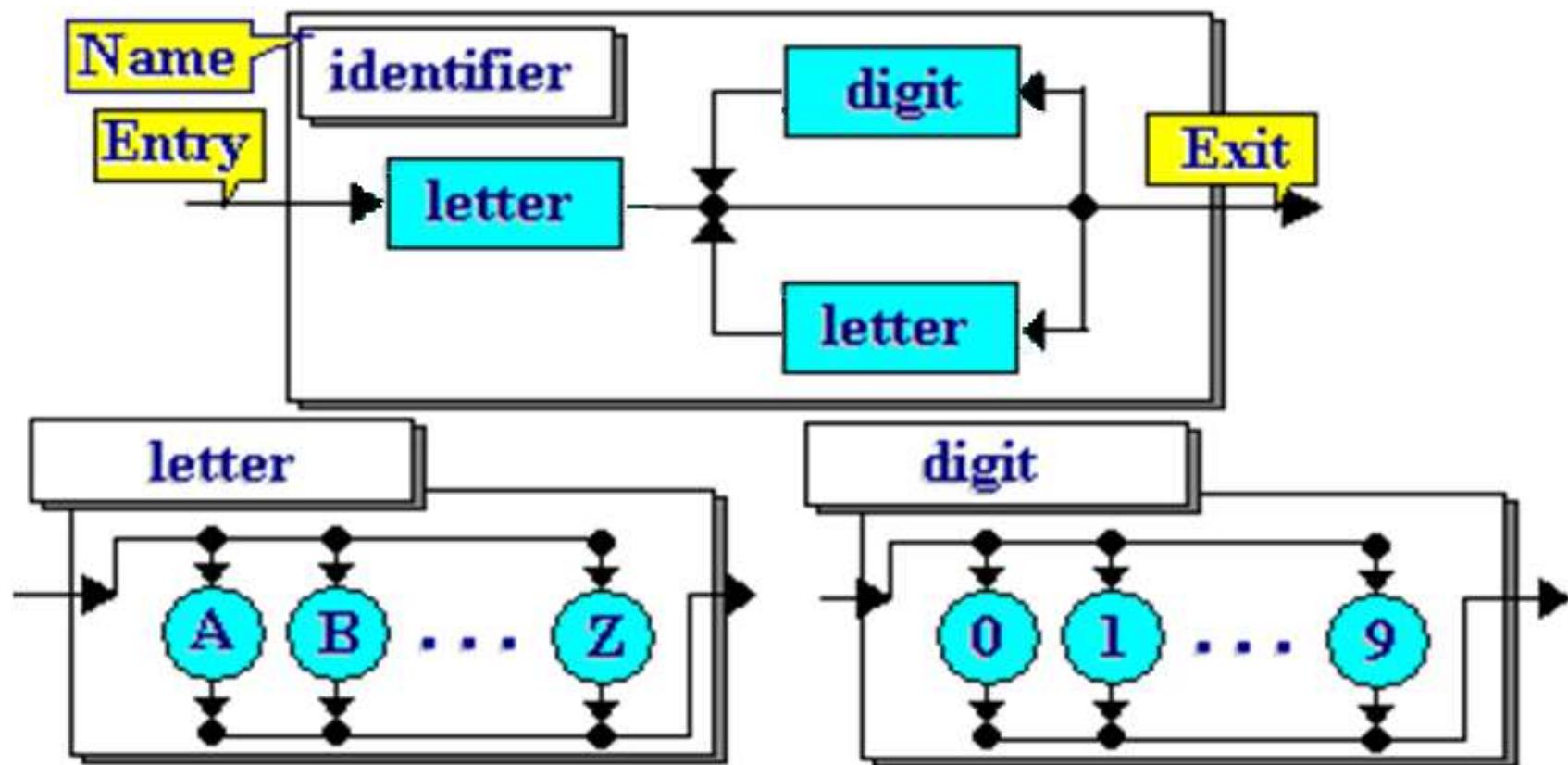


# Sơ đồ cú pháp

- Là công cụ để mô tả cú pháp của ngôn ngữ lập trình dưới dạng đồ thị
- Mỗi sơ đồ cú pháp là một đồ thị định hướng với lối vào và lối ra xác định.
- Mỗi sơ đồ cú pháp có một tên duy nhất

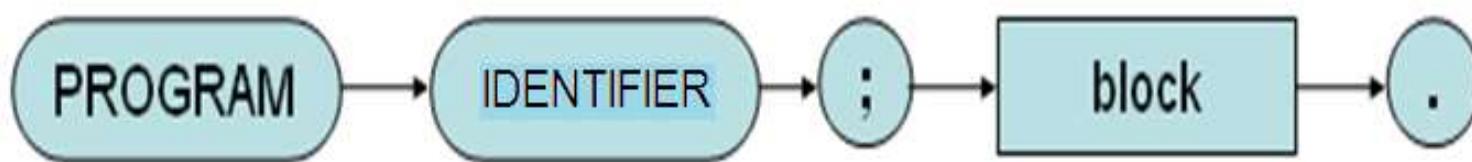


## Ví dụ một sơ đồ cú pháp

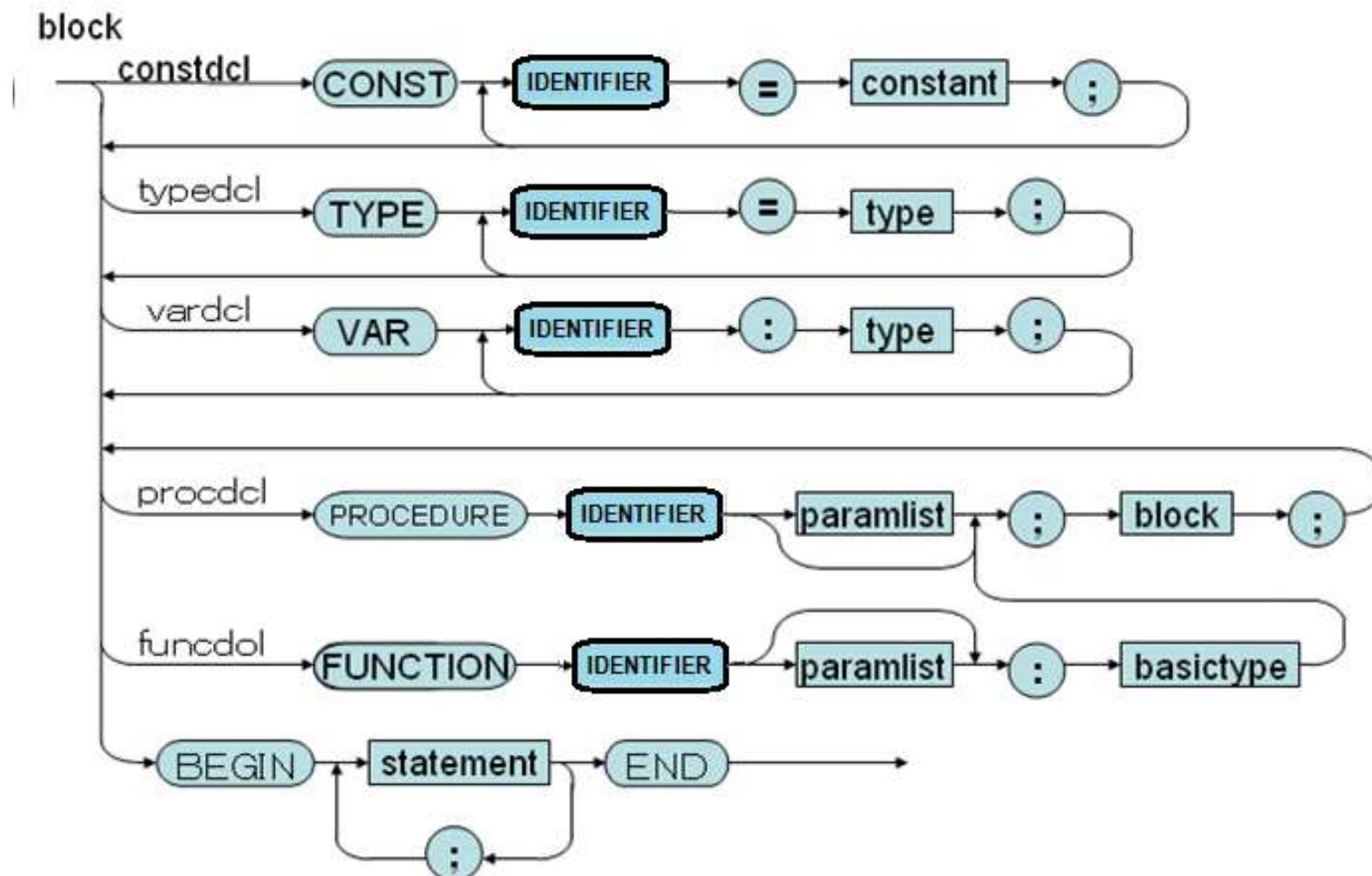


# Sơ đồ cú pháp của KPL (Tổng thể CT)

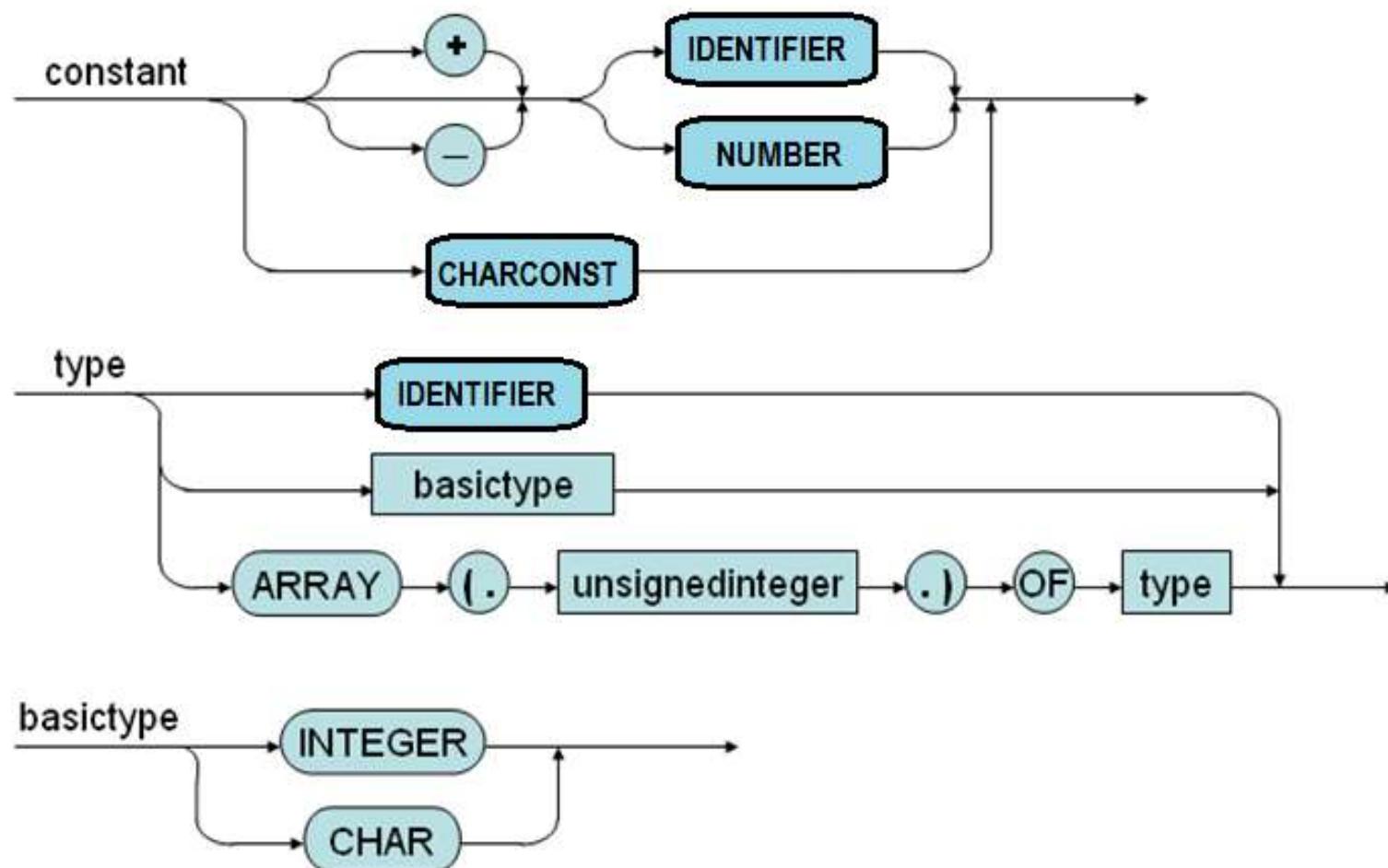
program



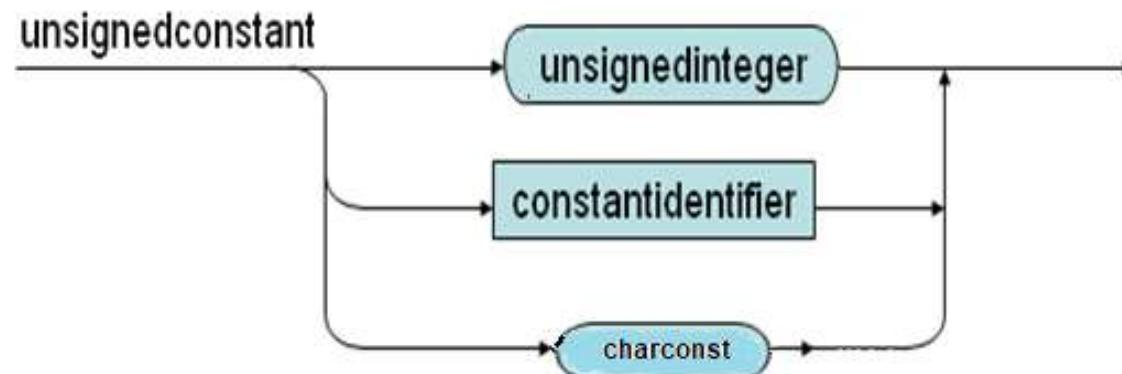
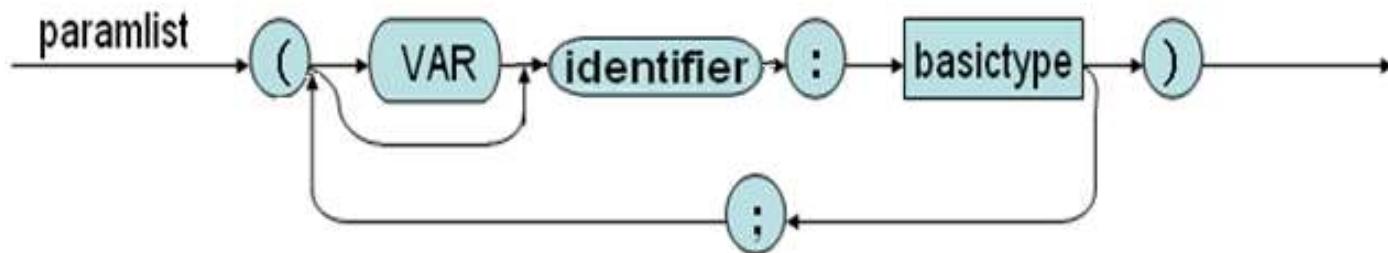
# Sơ đồ cú pháp của KPL (Khối)



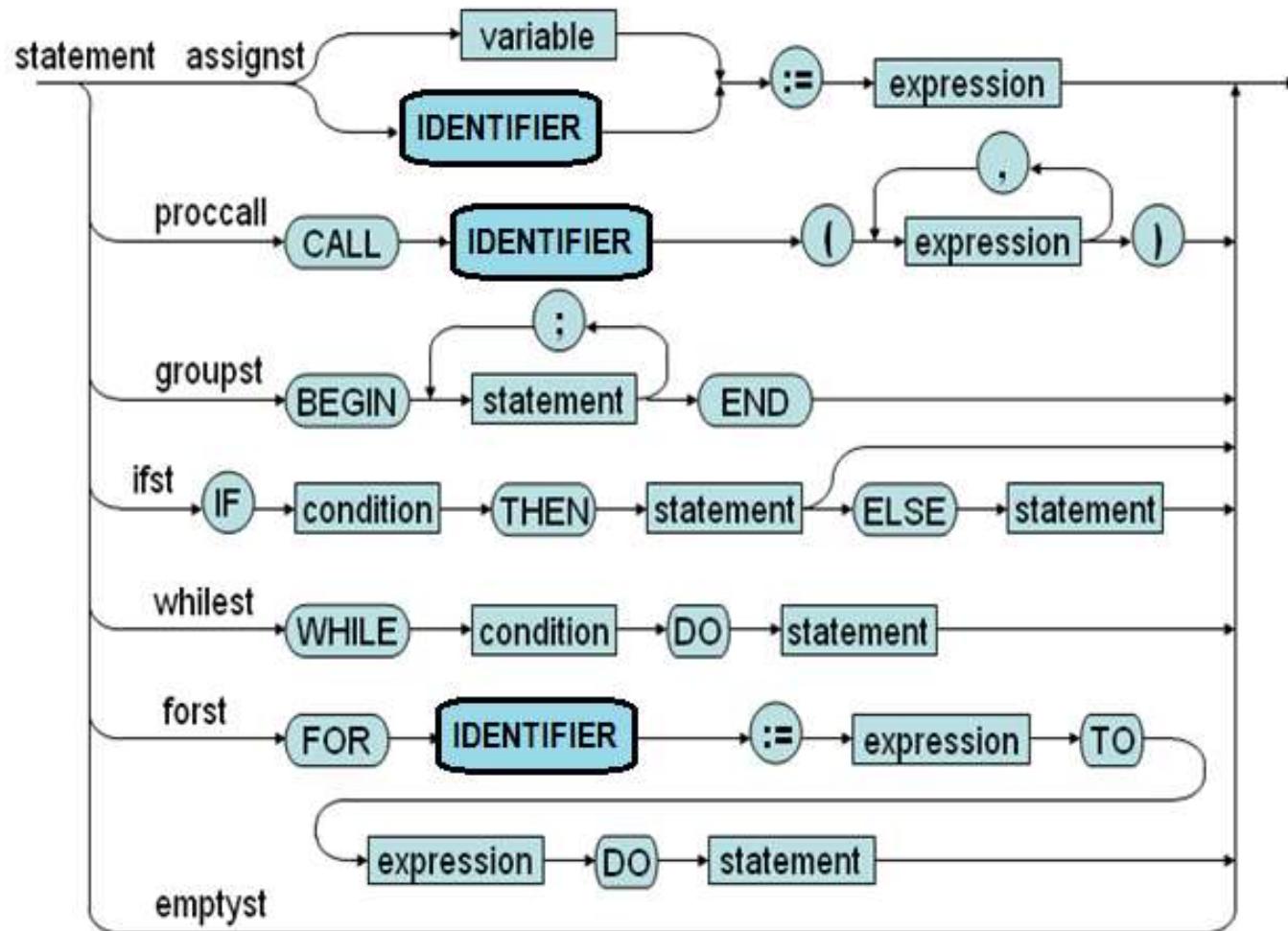
# Sơ đồ cú pháp của KPL (Khai báo)



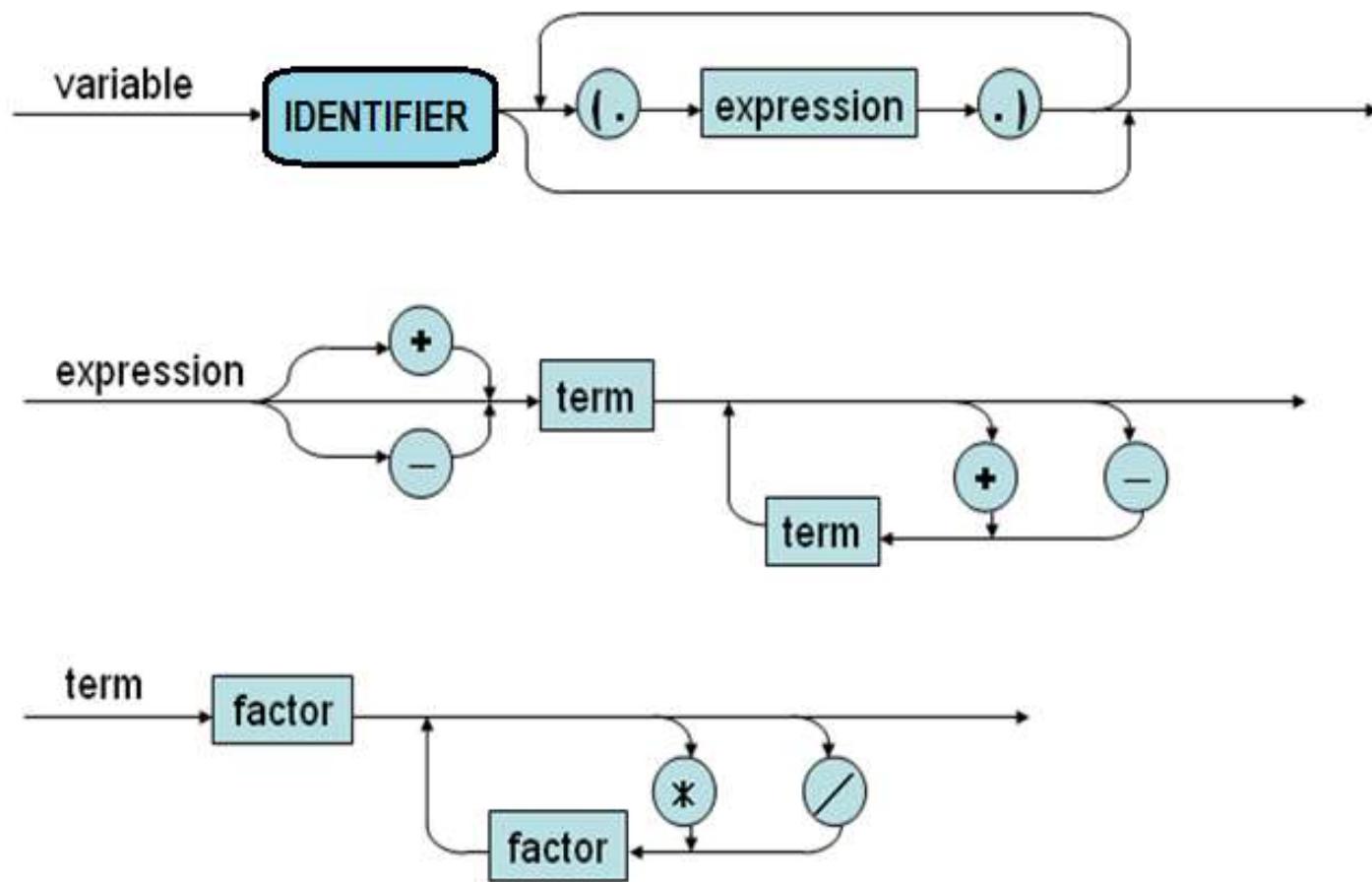
# Sơ đồ cú pháp của KPL(tham số, hằng không dấu)



# Sơ đồ cú pháp của KPL (lệnh)



# Sơ đồ cú pháp của KPL (biểu thức)



# Sơ đồ cú pháp của KPL (thừa số,điều kiện)

