**Bài 1.** Giả sử cú pháp cho phần khai báo của ngôn ngữ KPL2 như sau:

**<Block> ::= <Declarations> <Block2>**

**<Declarations>::=<ConstDecls2><Declarations>|<TypeDecls2><Declarations>| <VarDecls2><Declarations>|<SubDecls><Declarations>|ε**

**<Constdecls2> ::= KW\_CONST <ConstDecl> <ConstDecls>**

**<TypeDecls2> ::= KW\_TYPE <TypeDecl> <TypeDecls>**

**<VarDecl2> ::= KW\_VAR <VarDecl><VarDeclrs>**

**<SubDecls> ::= <FunDecl> <SubDecls>|<ProcDecl> <SubDecls>|ε**

**<FunDecl> ::= KW\_FUNCTION TK\_IDENT <Params> SB\_COLON <BasicType> SB\_SEMICOLON <Block> SB\_SEMICOLON**

**<ProcDeclb> ::= KW\_PROCEDURE TK\_IDENT <Params> SB\_SEMICOLON <Block> SB\_SEMICOLON**

**<Block2> ::= KW\_BEGIN <Statements> KW\_END**

1. Chứng minh các sản xuất với vế trái <Declarations> thỏa điều kiện LL(1).

FIRST(ConstDecls2) = {KW\_CONST}

FIRST(TypeDecls2) = {KW\_TYPE}

FIRST(VarDecls2) = {KW\_VAR}

FIRST(SubDecls) = {KW\_FUNCTION, KW\_PROCEDURE, ε}

FOLLOW(Declarations) = {KW\_BEGIN}

=> Thỏa ĐK LL(1)

1. Viết hàm void compileDeclarations(void) và hàm void compileTypeDeclrs2(void) để phân tích cú pháp phần khai báo.

FIRST(Declarations) = {KW\_CONST, KW\_TYPE, KW\_VAR, KW\_FUNCTION, KW\_PROCEDURE, ε}

FOLLOW(Declarations) = {KW\_BEGIN, $}

Ảnh có chứa văn bản

Mô tả được tạo tự động

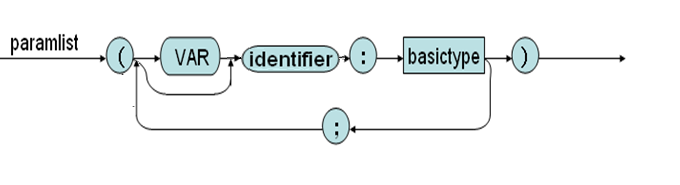
FIRST(TypeDecls2) = {KW\_TYPE}

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Cho biết sự khác biệt về cú pháp của phần khai báo trong ngôn ngữ KPL và KPL2

**Bài 2.** Viết hàm **void CompileParamlist(void)** để phân tích cú pháp danh sách tham số hình thức. Cú pháp tuân theo sơ đồ dưới đây:

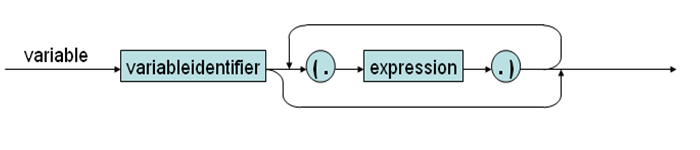
****

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

Giả sử hàm **compileBasictype** đã xây dựng

**Bài 3.** Viết hàm **void CompileVariable(void)** để phân tích cú pháp cho các biến, bao gồm cả biến đơn và biến mảng. Cú pháp tuân theo sơ đồ sau:

****

Ảnh có chứa văn bản

Mô tả được tạo tự động

Giả sử hàm **compileExpression** đã xây dựng.

**Bài 4:** Viêt chương trình bằng ngôn ngữ KPL thực hiện các công việc sau:

d) Tạo hàm WD nhận một số chỉ thứ của ngày trong tuần: 1, 2,….7. In ra

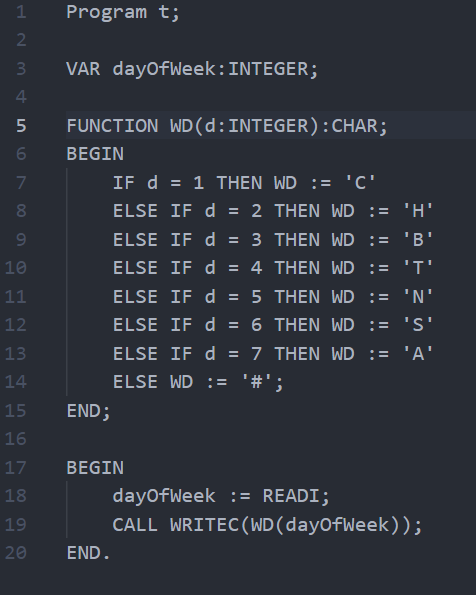
C,H,B,T,N,S,A tương ứng với các giá trị nhập vào là 1, 2, 3, 4, 5, 6, 7. In ra # nếu giá

trị nhập vào không thuộc danh sách trên.

e) Nhập một giá trị ngày trong tuần. In ra giá trị của hàm WD của ngày đó.

f) Hãy sửa đi một lệnh trong chương trình để gặp lỗi cú pháp

ERR\_INVALIDARGUMENTS. Giải thích vì sao xảy ra lỗi đó.



ERR\_INVALIDARGUMENTS: (Trong gọi hàm) Bỏ ) sau khi truyền tham số vào hàm





ERR\_INVALIDCONSTDECL: (Trong định nghĩa Const) Token đầu khác TK\_IDENT

ERR\_INVALIDTYPEDECL: (Trong định nghĩa Type) Token đầu khác TK\_IDENT

ERR\_INVALIDVARDECL: (Trong định nghĩa Var) Token đầu khác TK\_IDENT

ERR\_INVALIDSUBDECL: (Trong định nghĩa hàm) Một token bất kỳ sau Định nghĩa hàm, khác KW\_BEGIN

ERR\_INVALIDCHARCONSTANT: ‘abc’

ERR\_INVALIDCONSTANT: (Trong định nghĩa Const) TOKEN khác INTEGER, NUMBER sau dấu =

TOKEN KHÁC INTEGER, NUMBER sau dấu +, -

ERR\_INVALIDTYPE: (Trong định nghĩa Type) TOKEN Khác INTEGER, NUMBER, CHAR, TK\_IDENT sau dấu = (Hoặc bỏ)





ERR\_INVALIDBASICTYPE: Trong định nghĩa giá trị trả về của function hoặc kiểu dữ liệu array, khác INTEGER, CHAR

Ảnh có chứa văn bản

Mô tả được tạo tự động



ERR\_INVALIDPARAM: (Trong khai báo hàm) Thiếu dấu chấm phẩy ngăn cách tham số / Có chấm phẩy nhưng tiếp sau không có định nghĩa tham số khác







ERR\_INVALIDSTATEMENT: (Sau dấu chấm phẩy kết thúc 1 statement) 1 TOKEN Khác IDENT, CALL, IF, WHILE, FOR, BEGIN



ERR\_INVALIDCOMPARATOR (Trong IF, WHILE): Chỉ có 1 vế (không có dấu so sánh). VD: chỉ 1 số nguyên trong phần kiểm tra điều kiện



ERR\_INVALIDTERM: Theo sau một Factor với 1 token: := . ( (. (Khác + - \* / , so sánh, …) / Bỏ dấu





ERR\_INVALIDFACTOR: Phép toán thiếu 1 phần tử hoặc phần tử không hợp lệ.



Ảnh có chứa văn bản

Mô tả được tạo tự động

token.h

Ảnh có chứa văn bản

Mô tả được tạo tự động

scanner.c

#include <ctype.h>

Ảnh có chứa văn bản

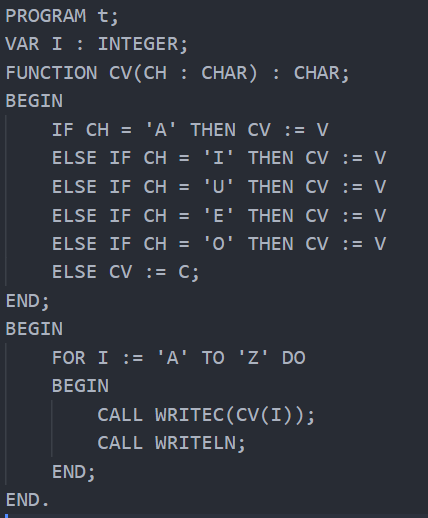
Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản, biên lai

Mô tả được tạo tự động



Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự độngẢnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

token.h

Ảnh có chứa văn bản

Mô tả được tạo tự động

scanner.c

#include <ctype.h>

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

FIRST(Params) = {SB\_LPAR}

Ảnh có chứa văn bản

Mô tả được tạo tự động

FIRST(Params1) = FIRST(Param) + {ε}

= {KW\_VAR, TK\_IDENT, ε}

FOLLOW(Params1) = {SB\_RPAR}

Ảnh có chứa văn bản

Mô tả được tạo tự động

Before: PROCEDURE A; BEGIN END;

After: PROCEDURE A(); BEGIN END;

Ảnh có chứa văn bản

Mô tả được tạo tự động

VAR A : ARRAY(. 100 .) OF ARRAY(. 100 .) OF INTEGER;

I : INTEGER;

J : INTEGER;

N : INTEGER;

T : INTEGER;

Begin

N := READI;

FOR I := 1 TO N DO

FOR J := 1 TO N DO

A(.I.)(.J.) := READI;

T := 1;

FOR I := 1 TO N DO

FOR J := 1 TO N DO

IF I > J THEN

IF A(.I.)(.J.) != 0 THEN

T := 0;

CALL WRITEI(T);

End.

Ảnh có chứa văn bản, bảng trắng

Mô tả được tạo tự động

FIRST(S) = {SB\_LPAR}

FIRST(S1) = FIRST(E) + E

FOLLOW(S1) = {SB\_RPAR}

FIRST(S2) = {SB\_COMMA, E}

FOLLOW(S2) = FOLLOW(S1) = {SB\_RPAR}