

# Báo cáo LAB02

## IT3323 Mã lớp 161269

### Bài 2 : Phân tích cú pháp

Họ và tên: Dương Công Thuyết

MSSV: 20225932

#### I. Giải thích logic code đã implement

##### 1. Type Declarations (Khai báo kiểu)

compileTypeDecls()

Chức năng: Xử lý nhiều khai báo kiểu liên tiếp

Hoạt động: Vòng lặp while kiểm tra xem token tiếp theo có phải là TK\_IDENT không, nếu có thì gọi compileTypeDecl() để xử lý từng khai báo kiểu

Ví dụ: TYPE myArray = ARRAY [10] OF INTEGER; myChar = CHAR;

compileTypeDecl()

Chức năng: Xử lý một khai báo kiểu đơn

Hoạt động:

Đọc tên kiểu (TK\_IDENT)

Đọc dấu "=" (SB\_EQ)

Gọi compileType() để xử lý định nghĩa kiểu

Đọc dấu ";" (SB\_SEMICOLON)

Cú pháp: identifier = Type;

##### 2. Variable Declarations (Khai báo biến)

compileVarDecls()

Chức năng: Xử lý nhiều khai báo biến liên tiếp

Hoạt động: Tương tự compileTypeDecls(), vòng lặp while kiểm tra TK\_IDENT và gọi compileVarDecl()

Ví dụ: VAR x : INTEGER; y : CHAR;

compileVarDecl()

Chức năng: Xử lý một khai báo biến đơn

Hoạt động:

Đọc tên biến (TK\_IDENT)

Đọc dấu ":" (SB\_COLON)

Gọi compileType() để xác định kiểu dữ liệu

Đọc dấu ";" (SB\_SEMICOLON)

Cú pháp: identifier : Type;

##### 3. Subroutine Declarations (Khai báo chương trình con)

compileSubDecls()

Chức năng: Xử lý tất cả các khai báo hàm (FUNCTION) và thủ tục (PROCEDURE)

Hoạt động: Vòng lặp while kiểm tra token là KW\_FUNCTION hoặc KW\_PROCEDURE, sau đó gọi hàm tương ứng

**Điểm đặc biệt:** Cho phép khai báo nhiều function/procedure liên tiếp  
compileFuncDecl()

**Chức năng:** Xử lý khai báo hàm (có giá trị trả về)

**Hoạt động:**

Đọc từ khóa FUNCTION

Đọc tên hàm (TK\_IDENT)

Nếu có "(", đọc danh sách tham số và đóng ")"

Đọc dấu ":" (SB\_COLON)

Gọi compileBasicType() để xác định kiểu trả về

Đọc dấu ";" (SB\_SEMICOLON)

Gọi compileBlock() để xử lý thân hàm

Đọc dấu ";" kết thúc

Cú pháp: FUNCTION name(params) : Type; Block;

compileProcDecl()

**Chức năng:** Xử lý khai báo thủ tục (không có giá trị trả về)

**Hoạt động:** Tương tự compileFuncDecl() nhưng không có phần khai báo kiểu trả về

Cú pháp: PROCEDURE name(params); Block;

#### 4. Constants (Hằng số)

compileUnsignedConstant()

**Chức năng:** Xử lý hằng số không dấu

**Hoạt động:** Sử dụng switch-case để kiểm tra và đọc:

TK\_NUMBER: số nguyên

TK\_IDENT: tên hằng đã định nghĩa trước

TK\_CHAR: ký tự

Nếu không phải các loại trên → báo lỗi ERR\_INVALID\_CONSTANT

compileConstant()

**Chức năng:** Xử lý hằng số có thể có dấu (+/-)

**Hoạt động:**

Nếu có SB\_PLUS hoặc SB\_MINUS → đọc dấu rồi gọi compileConstant2()

Nếu là TK\_CHAR → đọc trực tiếp

Ngược lại → gọi compileConstant2()

Ví dụ: +5, -10, 'a', MAX\_VALUE

compileConstant2()

**Chức năng:** Xử lý phần số hoặc identifier của hằng

**Hoạt động:** Kiểm tra và đọc TK\_IDENT hoặc TK\_NUMBER, báo lỗi nếu không phải

#### 5. Basic Types (Kiểu cơ bản)

compileBasicType()

**Chức năng:** Xử lý các kiểu dữ liệu cơ bản

**Hoạt động:** Switch-case kiểm tra:

KW\_INTEGER → kiểu số nguyên

KW\_CHAR → kiểu ký tự

Nếu không phải → báo lỗi ERR\_INVALIDBASICTYPE

## 6. Parameters (Tham số)

compileParams()

Chức năng: Xử lý danh sách tham số

Hoạt động: Gọi compileParam() cho tham số đầu tiên, sau đó gọi compileParams2() cho các tham số còn lại  
compileParams2()

Chức năng: Xử lý các tham số bổ sung

Hoạt động: Đệ quy - nếu gặp ";" thì đọc tham số tiếp theo và gọi lại chính nó

Cú pháp: param1 : Type; param2 : Type; ...

compileParam()

Chức năng: Xử lý một tham số đơn

Hoạt động:

Nếu bắt đầu bằng TK\_IDENT: tham số truyền giá trị name : Type

Nếu bắt đầu bằng KW\_VAR: tham số truyền tham chiếu VAR name : Type

Ngược lại → báo lỗi ERR\_INVALIDPARAM

## 7. Statements (Các câu lệnh)

compileAssignSt()

Chức năng: Xử lý câu lệnh gán

Hoạt động:

Đọc tên biến (TK\_IDENT)

Gọi compileIndexes() để xử lý chỉ số mảng (nếu có)

Đọc toán tử gán ":=" (SB\_ASSIGN)

Gọi compileExpression() để tính giá trị về phải

Cú pháp: variable := expression hoặc array[index] := expression

compileCallSt()

Chức năng: Xử lý câu lệnh gọi thủ tục/hàm

Hoạt động:

Đọc từ khóa CALL

Đọc tên thủ tục (TK\_IDENT)

Nếu có "(", đọc danh sách đối số và đóng ")"

Cú pháp: CALL procedureName hoặc CALL procedureName(args)

compileGroupSt()

Chức năng: Xử lý khối lệnh (compound statement)

Hoạt động:

Đọc BEGIN

Gọi compileStatements() để xử lý các câu lệnh bên trong

Đọc END

Cú pháp: BEGIN statements END

compileWhileSt()

Chức năng: Xử lý vòng lặp while

Hoạt động:

Đọc từ khóa WHILE

Gọi compileCondition() để xử lý điều kiện

Đọc từ khóa DO

Gọi compileStatement() để xử lý câu lệnh lặp

Cú pháp: WHILE condition DO statement

compileForSt()

Chức năng: Xử lý vòng lặp for

Hoạt động:

Đọc từ khóa FOR

Đọc biến đếm (TK\_IDENT)

Đọc ":" (SB\_ASSIGN)

Gọi compileExpression() cho giá trị bắt đầu

Đọc từ khóa TO

Gọi compileExpression() cho giá trị kết thúc

Đọc từ khóa DO

Gọi compileStatement() cho thân vòng lặp

Cú pháp: FOR i := start TO end DO statement

## 8. Arguments (Đối số)

compileArguments()

Chức năng: Xử lý danh sách đối số khi gọi hàm/thủ tục

Hoạt động: Gọi compileExpression() cho đối số đầu tiên, sau đó

compileArguments2() cho các đối số còn lại

compileArguments2()

Chức năng: Xử lý các đối số bổ sung

Hoạt động: Đệ quy - nếu gặp "," thì đọc biểu thức tiếp theo

Cú pháp: arg1, arg2, arg3, ...

## 9. Conditions (Điều kiện)

compileCondition()

Chức năng: Xử lý điều kiện so sánh

Hoạt động: Gọi compileExpression() cho vế trái, sau đó compileCondition2()

cho toán tử và vế phải

compileCondition2()

Chức năng: Xử lý toán tử so sánh và vế phải

Hoạt động: Switch-case kiểm tra các toán tử:

SB\_EQ: bằng (=)

SB\_NEQ: khác (<>)

SB\_LT: nhỏ hơn (<)

SB\_LE: nhỏ hơn hoặc bằng (<=)

SB\_GT: lớn hơn (>)

SB\_GE: lớn hơn hoặc bằng (>=)

Sau đó gọi compileExpression() cho vế phải

Cú pháp: expression comparator expression

## 10. Expressions (Biểu thức)

compileExpression()

Chức năng: Xử lý biểu thức với dấu (+/-) ở đầu (optional)

Hoạt động:

Nếu có SB\_PLUS hoặc SB\_MINUS → đọc dấu rồi gọi compileExpression2()

Ngược lại → gọi trực tiếp compileExpression2()

Ví dụ: +5, -x, a + b

compileExpression2()

Chức năng: Xử lý term và các phép cộng/trừ tiếp theo

Hoạt động: Gọi compileTerm() cho term đầu tiên, sau đó

compileExpression3() cho các phép toán tiếp theo

compileExpression3() (đã có sẵn)

Chức năng: Xử lý các phép cộng/trừ liên tiếp (đệ quy trái)

Hoạt động: Nếu gặp + hoặc -, đọc toán tử, gọi compileTerm(), rồi gọi lại chính nó

## 11. Factor (Thừa số)

compileFactor()

Chức năng: Xử lý các thừa số cơ bản trong biểu thức

Hoạt động: Switch-case kiểm tra:

TK\_NUMBER: Đọc số nguyên

TK\_CHAR: Đọc ký tự

TK\_IDENT: Đọc identifier, sau đó:

Nếu có "(": gọi hàm với đối số → compileArguments()

Ngược lại: biến hoặc mảng → compileIndexes()

SB\_LPAR: Biểu thức trong ngoặc → đọc "(", compileExpression(), đọc ")"

Ngược lại → báo lỗi ERR\_INVALIDFACTOR

Ví dụ: 5, 'a', x, arr[i], func(a,b), (x+y)

II. Kết quả chạy với chương trình không lỗi

## Example1:

```
/* @copyright (c) 2008, Hanoi University of Technology
 * @version 1.0
 */
#include <stdlib.h>
#include <stdio.h>
#include "reader.h"
#include "scanner.h"
#include "parser.h"
#include "error.h"

Token *currentToken;
Token *lookAhead;
LookAhead = getValidToken();
free(tmp);

void scan(void)
{
    Token *tmp = currentToken;
    currentToken = lookAhead;
    LookAhead = getValidToken();
    free(tmp);
}

void eat(TokenType tokenType)
{
    if (lookAhead->tokenType == tokenType)
    {
        printToken(lookAhead);
        scan();
    }
}

Subroutines parsed ....
2-1:Kw BEGIN
3-1:Kw END
Block parsed!
3-5:SB_PERIOD
Program parsed!
```

duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/example1.kpl

Program parsed!  
Parser 3: Program ...  
1-1:Kw PROGRAM  
1-9:TK IDENT(example1)  
1-17:SR\_SEMICOLON  
Parsing a Block ...  
Parsing subroutines ...  
Subroutines parsed ....  
2-1:Kw BEGIN  
3-1:Kw END  
Block parsed!  
3-5:SB\_PERIOD  
Program parsed!

Các example còn lại được ghi trong file Ketqua.pdf

### III. Các trường hợp có lỗi

#### 1. Lỗi ERR\_END\_OF\_COMMENT

Lỗi ERR\_END\_OF\_COMMENT ở dòng 4: Comment bắt đầu bằng (\*) nhưng không có dấu đóng (\*)

Đoạn code lỗi: " This is an unclosed comment..." - thiếu dấu \*) kết thúc

```
PROGRAM P1;
BEGIN
/* This is an unclosed comment that should trigger
END.
```

duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/1.kpl

Parser a Program ...
1-1:Kw PROGRAM
1-9:TK IDENT(P1)
1-11:SR\_SEMICOLON
Parsing a Block ...
Parsing subroutines ...
Subroutines parsed ...
2-1:Kw BEGIN
5-1:End of comment expected!

## 2. Lỗi ERR\_IDENT\_TOO\_LONG

Lỗi ERR\_IDENT\_TOO\_LONG ở dòng 3: Tên biến có 63 ký tự, vượt quá giới hạn cho phép

## Đoạn code lỗi:

"thisisaverylongidentifiername that exceedsthemaximumallowedlength"

The screenshot shows a terminal window with the following output:

```
duongcongthuyet@Duongs-MacBook-Pro PT_CPIcompleted % ./parser Test.2.kpl
Parsing a program ...
1-1:KW_PROGRAM
1-9:TK_IDENT(P2)
1-11:SB_SEMICOLON
Parsing a block ...
2-1:KW_BLOCK
2-5:Identification too long!
duongcongthuyet@Duongs-MacBook-Pro PT_CPIcompleted %
```

### 3. Lỗi ERR\_NUMBER\_TOO\_LONG

Đoạn code lỗi: "max = 99999999999999999999" - số có 20 chữ số

```

PT_CPincompleted
1.kpl 2.kpl
Test > 2.kpl
1 PROGRAM P2;
2 VAR thisisaverylongidentifiernamethatexceedsthemaximumallowablelength : INTEGER;
3 BEGIN
4   thisisaverylongidentifiernamethatexceedsthemaximumallowablelength := 10;
5 END.
6

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

- duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/3.kpl
  - Parsing a Program ....
  - 1-1:KW\_PROGRAM
  - 1-9:TK\_IDENTIFIER(P3)
  - 1-11:SB\_COLON
  - Parsing a Block ....
  - 2-1:KW\_VAR
  - 2-5:TK\_IDENTIFIER(x)
  - 2-7:SB\_COLON
  - 2-9:KW\_INTEGER
  - 2-10:SB\_SEMICOLON
  - Parsing subroutines ....
  - Subroutines parsed ....
  - 3-1:KW\_BEGIN
  - Parsing an assign statement ....
  - 4-3:TK\_IDENTIFIER(x)
  - 4-6:SB\_ASSIGN
  - 4-8:Value of Integer number exceeds the range!

duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted %

#### 4. Lỗi ERR\_INVALIDCHARCONSTANT

Lỗi ERR\_INVALIDCHARCONSTANT ở dòng 5: Hằng ký tự 'a; thiếu dấu ngoặc đơn đóng

Đoạn code lỗi: "c := 'a;" - phải là "c := 'a';"

```

PT_CPincompleted
1.kpl 4.kpl 5.kpl
Test > 4.kpl
1 PROGRAM P4;
2 VAR c : CHAR;
3 BEGIN
4   c := 'a;
5 END.
6

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

- duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/4.kpl
  - Parsing a Program ....
  - 1-1:KW\_PROGRAM
  - 1-9:TK\_IDENTIFIER(P4)
  - 1-11:SB\_COLON
  - Parsing a Block ....
  - 2-1:KW\_VAR
  - 2-5:TK\_IDENTIFIER(c)
  - 2-7:SB\_COLON
  - 2-9:KW\_CHAR
  - 2-10:SB\_SEMICOLON
  - Parsing subroutines ....
  - Subroutines parsed ....
  - 3-1:KW\_BEGIN
  - Parsing an assign statement ....
  - 4-3:TK\_IDENTIFIER(c)
  - 4-6:SB\_ASSIGN
  - 4-8:Invalid Const char!

duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted %

## 5. Lỗi ERR\_INVALID\_SYMBOL

Lỗi ERR\_INVALID\_SYMBOL ở dòng 4: Ký hiệu @ không hợp lệ trong KPL

Đoạn code lỗi: "x := 5 @ 3;" - ký tự @ không thuộc bộ ký hiệu của ngôn ngữ

The screenshot shows a terminal window with the following text:

```
duongcongthuyet@Duongs-MacBook-Pro PT_CPICompleted % ./parser Test/4.kpl
4-6:SB ASCII
4-8:Invalid constant!
● duongcongthuyet@Duongs-MacBook-Pro PT_CPICompleted % ./parser Test/5.kpl
Parsin a Program ....
1-1:KW_PROGRAM
1-11:SB_SETCOLON
Parsin a Block ....
2-1:KW_VAR
2-5:TK_IDENT(x)
2-7:SB_COLON
2-9:TK_ASSIGN
2-16:SB_SETCOLON
Parsin subroutines ....
Subroutines parsed ....
3-1:KW_BEGIN
Parsin an assign statement ....
4-3:TK_ASSIGN(x)
4-6:SB_SETCOLON
Parsin an expression
4-8:TK_NUMBER(10)
4-11:Invalid symbol!
```

The terminal window is part of a larger interface with an Explorer sidebar showing files like 1.kpl, 2.kpl, and various .kpl files in the PT\_CPICompleted project.

## 6. Lỗi ERR\_INVALID\_CONSTANT

Lỗi ERR\_INVALID\_CONSTANT ở dòng 3: Khai báo CONST thiếu giá trị sau dấu =  
Đoạn code lỗi: "CONST x = ;" - thiếu giá trị hằng số giữa = và ;

```

1 PROGRAM P2;
2 VAR thisisaverylongidentifiernamethatexceedsthemaximumallowablelength : INTEGER;
3 BEGIN
4   thisisaverylongidentifiernamethatexceedsthemaximumallowablelength := 10;
5 END.
6

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

- duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/6.kpl
  - Parsing a Program ...
  - 1-1:KW\_PROGRAM
  - 1-9:TK\_IDENT(P6)
  - 1-11:SB\_SEMICOLON
  - Parsing a Block ...
  - 2-1:KW\_CONST
  - 2-7:TK\_IDENT(x)
  - 2-9:SB\_EQ
  - 2-11:Invalid constant!

## 7. Lỗi ERR\_INVALIDTYPE

Lỗi ERR\_INVALIDTYPE ở dòng 3: Khai báo ARRAY sai cú pháp, thiếu [size] OF

Đoạn code lỗi: "arr = ARRAY INTEGER" - phải là "arr = ARRAY [10] OF INTEGER"

```

1 PROGRAM P7;
2 TYPE mytype = ARRAY(. 10 .) OF;
3 BEGIN
4 END.
5

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

- duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/7.kpl
  - Parsing a Program ...
  - 1-1:KW\_PROGRAM
  - 1-9:TK\_IDENT(P7)
  - 1-11:SB\_SEMICOLON
  - Parsing a Block ...
  - 2-1:KW\_TYPE
  - 2-6:TK\_IDENT(mytype)
  - 2-13:SB\_EQ
  - 2-15:KW\_ARRAY
  - 2-20:SB\_SEL
  - 2-23:SB\_NUMBER(10)
  - 2-27:SB\_REL
  - 2-29:KW\_OF
  - 2-31:Invalid type!

## 8. Lỗi ERR\_INVALIDBASICTYPE

Lỗi ERR\_INVALIDBASICTYPE ở dòng 3: FLOAT không phải kiểu cơ bản hợp lệ

Đoạn code lỗi: "FUNCTION test : FLOAT;" - chỉ có INTEGER và CHAR là hợp lệ

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists various KPL files (1.kpl, 2.kpl, 3.kpl, etc.) and some C/C++ header files (charcode.c, error.h). The main editor area displays the content of the 8.kpl file:

```
Test > 8.kpl
1 PROGRAM PB;
2 FUNCTION test : FLOAT;
3 BEGIN
4 END;
5 BEGIN
6 END.
```

The terminal tab at the bottom shows the command `./parser Test/8.kpl` being run, with the output indicating a parsing error at line 2, column 9, due to an invalid basic type:

```
duongcongthuyet@Duongs-MacBook-Pro PT_CPIcompleted % ./parser Test/8.kpl
Parsing a Program ....
1-1:KW PROGRAM
1-9:TK IDENT(P8)
1-11:SB SEMICOLON
Parsing a Block ....
Parsing subroutines ....
2-1:KW FUNCTION
2-10:TK IDENT(test)
2-15:SB_COLON
2-17:Invalid basic type!
```

## 9. Lỗi ERR\_INVALIDPARAM

Lỗi ERR\_INVALIDPARAM ở dòng 3: Tham số thiếu dấu : và kiểu dữ liệu

Đoạn code lỗi: "test(x y)" - phải là "test(x : INTEGER; y : INTEGER)"

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists various KPL files (1.kpl, 2.kpl, 3.kpl, etc.) and some C/C++ header files (charcode.c, error.h). The main editor area displays the content of the 9.kpl file:

```
Test > 9.kpl
1 PROGRAM P9;
2 PROCEDURE myproc(5 : INTEGER);
3 BEGIN
4 END;
5 BEGIN
6 END.
```

The terminal tab at the bottom shows the command `./parser Test/9.kpl` being run, with the output indicating a parsing error at line 2, column 11, due to an invalid parameter:

```
duongcongthuyet@Duongs-MacBook-Pro PT_CPIcompleted % ./parser Test/9.kpl
Parsing a Program ....
1-1:KW PROGRAM
1-9:TK IDENT(P9)
1-11:SB_SEMICOLON
Parsing a Block ....
Parsing subroutines ....
2-1:KW PROCEDURE
2-11:TK_IDENT(myproc)
2-17:SB_LPAR
2-18:Invalid parameter!
```

## 10. Lỗi ERR\_INVALIDSTATEMENT

Lỗi ERR\_INVALIDSTATEMENT ở dòng 4: RETURN không phải là câu lệnh hợp lệ

Đoạn code lỗi: "RETURN 5;" - KPL không có câu lệnh RETURN

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists several KPL files (1.kpl, 2.kpl, 3.kpl, 4.kpl, 5.kpl, 6.kpl, 7.kpl, 8.kpl, 9.kpl, 10.kpl) and other files like example1.kpl, charcode.c, error.c, etc. The main editor area shows a file named 10.kpl with the following content:

```
PROGRAM P10;
BEGIN
3 := 10; (* Invalid statement - missing left-hand side *)
END.
```

The line "3 := 10; (\* Invalid statement - missing left-hand side \*)" is highlighted in red, indicating an error. Below the editor is the Problems panel, which displays the following log:

- duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/10.kpl
- Parsing a Program ....
- 1-1:Kw PROGRAM
- 1-9:Tk IDENT(P10)
- 1-12:Sb SEMICOLON
- Parsing a Block ....
- Parsing subroutines ....
- Subroutines parsed ....
- 2-1:Kw BEGIN
- 3-4:Invalid statement!

## 11. Lỗi ERR\_INVALIDARGUMENTS

Lỗi ERR\_INVALIDARGUMENTS ở dòng 8: Danh sách đối số thiếu biểu thức sau dấu phẩy

Đoạn code lỗi: "CALL test(5, );" - thiếu đối số thứ 2 sau dấu phẩy

```

1 PROGRAM P11;
2 PROCEDURE myproc(x : INTEGER);
3 BEGIN
4 END;
5 BEGIN
6 CALL myproc(10, ); (* Invalid arguments - missing second argument *)
7 END.
8

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/11.kpl

2-22:KW\_INTEGER  
2-29:SB\_RPAR  
2-30:SR\_SEMICOLON  
Parsing a Block ....  
Parsing subroutines ....  
Subroutines parsed ....  
3-1:KW\_BEGIN  
4-1:KW\_END  
Block parsed!  
4-4:TK\_SEMICOLON  
Procedure parsed ....  
Subroutines parsed ....  
5-1:KW\_BEGIN  
Parsing a call statement ....  
6-3:TK\_CALL  
6-4:TK\_IDENT(myproc)  
6-14:SR\_LPAR  
Parsing an expression  
6-15:TK\_NUMBER(10)  
Expression parsed  
6-16:TK\_SEMICOLON  
6-19:Invalid arguments!

## 12. Lỗi ERR\_INVALIDCOMPARATOR

Lỗi ERR\_INVALIDCOMPARATOR ở dòng 5: Toán tử >> không phải toán tử so sánh hợp lệ

Đoạn code lỗi: "IF x >> 5" - chỉ có =, <, <=, >, >= là hợp lệ

```

1 PROGRAM P12;
2 VAR x : INTEGER;
3 BEGIN
4 IF x + 10 THEN
5 END;
6 END.
7

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

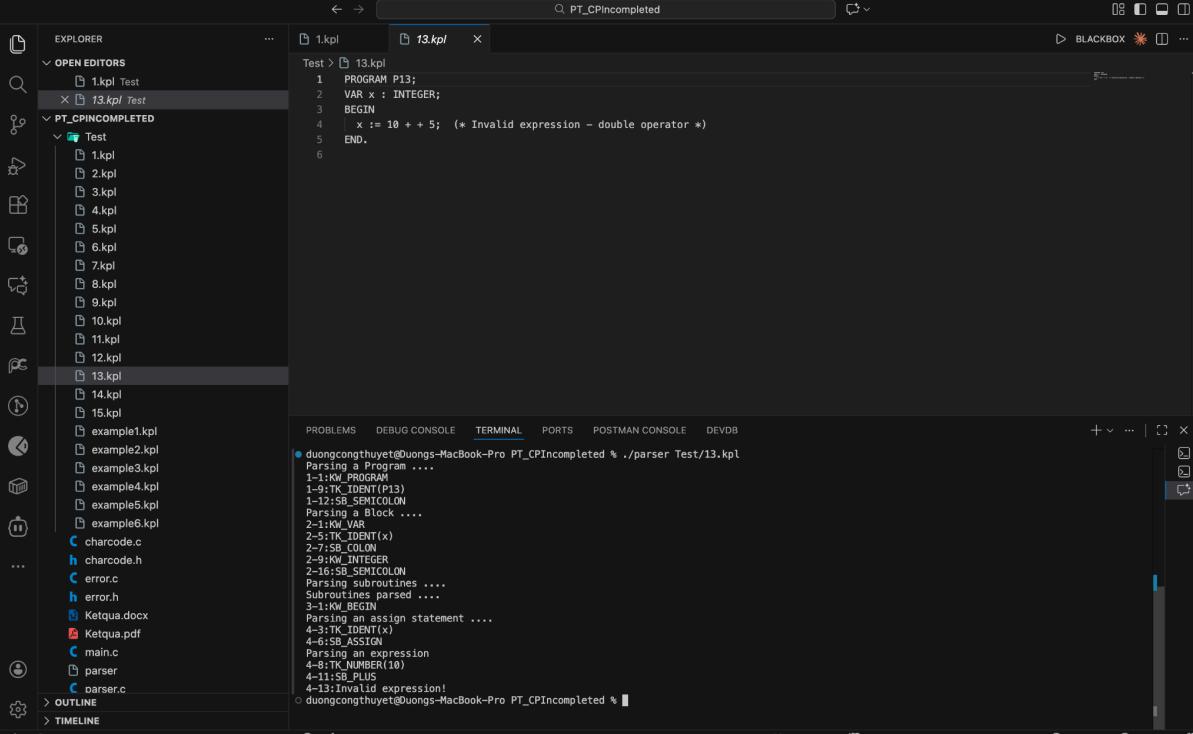
duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/12.kpl

1-1:KW\_PROGRAM  
1-3:TK\_IDENTIFIER(P12)  
1-12:SR\_SEMICOLON  
Parsing a Block ....  
2-1:KW\_VAR  
2-5:TK\_IDENT(x)  
2-7:SB\_COLON  
2-9:TK\_INTEGER  
2-16:SR\_SEMICOLON  
Parsing subroutines ....  
Subroutines parsed ....  
3-1:KW\_BEGIN  
Parsing an if statement ....  
4-3:TK\_IF  
Parsing an expression  
4-6:TK\_IDENT(x)  
4-8:SB\_PLUS  
4-10:TK\_NUMBER(10)  
Expression parsed  
4-13:Invalid comparator!

## 13. Lỗi ERR\_INVALIDEXPRESSION

Lỗi ERR\_INVALIDEXPRESSION ở dòng 5: Biểu thức thiếu toán hạng sau dấu +

Đoạn code lỗi: "x := 5 + ;" - thiếu term/factor sau toán tử cộng



```
1 PROGRAM P13;
2 VAR x : INTEGER;
3 BEGIN
4   x := 10 + + 5; (* Invalid expression - double operator *)
5 END.
```

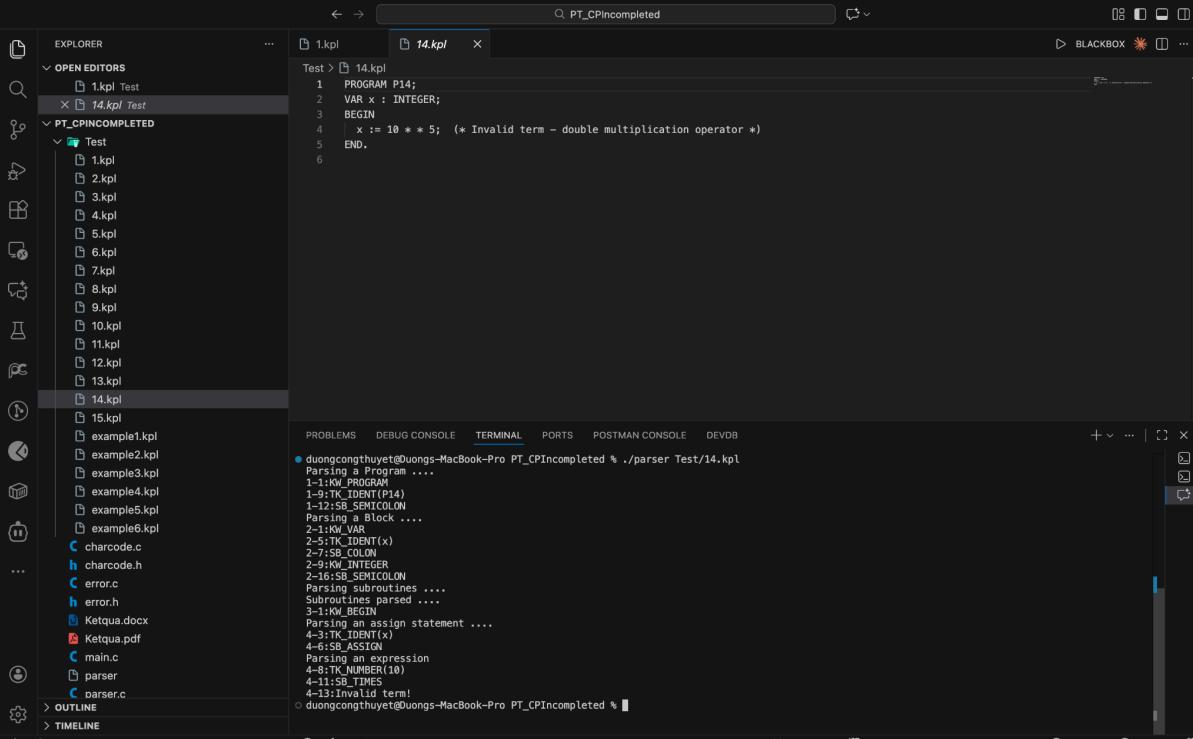
PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

- duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/13.kpl  
Parsing a Program ....  
1-1:KW\_PROGRAM  
1-9:TK\_IDENT(P13)  
1-12:SB\_SEMICOLON  
Parsing a Block ....  
2-1:KW\_VAR  
2-5:TK\_IDENT(x)  
2-7:SB\_COLON  
2-9:KW\_INTEGER  
2-16:SB\_SEMICOLON  
Parsing Subroutines ....  
Subroutines parsed ....  
3-1:KW\_BEGIN  
Parsing an assign statement ....  
4-3:TK\_IDENT(x)  
4-6:SB\_ASSIGN  
Parsing an expression  
4-8:TK\_NUMBER(10)  
4-11:SB\_PLUS  
4-13:Invalid expression!

## 14. Lỗi ERR\_INVALIDTERM

Lỗi ERR\_INVALIDTERM ở dòng 5: Term thiếu factor sau toán tử \*

Đoạn code lỗi: "x := 5 \* ;" - thiếu factor sau toán tử nhân



```
1 PROGRAM P14;
2 VAR x : INTEGER;
3 BEGIN
4   x := 10 * * 5; (* Invalid term - double multiplication operator *)
5 END.
```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE DEVDB

- duongcongthuyet@Duongs-MacBook-Pro PT\_CPincompleted % ./parser Test/14.kpl  
Parsing a Program ....  
1-1:KW\_PROGRAM  
1-9:TK\_IDENT(P14)  
1-12:SB\_SEMICOLON  
Parsing a Block ....  
2-1:KW\_VAR  
2-5:TK\_IDENT(x)  
2-7:SB\_COLON  
2-9:KW\_INTEGER  
2-16:SB\_SEMICOLON  
Parsing Subroutines ....  
Subroutines parsed ....  
3-1:KW\_BEGIN  
Parsing an assign statement ....  
4-3:TK\_IDENT(x)  
4-6:SB\_ASSIGN  
Parsing an expression  
4-8:TK\_NUMBER(10)  
4-11:SB\_TIMES  
4-13:Invalid term!

## 15. Lỗi ERR\_INVALIDFACTOR

Lỗi ERR\_INVALIDFACTOR ở dòng 5: BEGIN không phải là factor hợp lệ

Đoạn code lỗi: "x := BEGIN;" - factor chỉ có thể là số, ký tự, biến, hàm, hoặc (expression)

The screenshot shows a code editor interface with the following details:

- Explorer:** Shows files like 1.kpl, 15.kpl, and various .kpl files from 2 to 14.
- Open Editors:** Displays two files: 1.kpl (Test) and 15.kpl (Test).
- Editor Content (15.kpl):**

```
1 PROGRAM P15;
2 VAR x : INTEGER;
3 BEGIN
4   x := ( ); (* Invalid factor - empty parentheses *)
5 END.
```
- Terminal:** Shows the output of the parser command:

```
duongcongthuyet@Duongs-MacBook-Pro PT_CPincompleted % ./parser Test/15.kpl
Parsing a Program ....
1:1:KW_PROGRAM
1-9:TK_IDENT(P15)
1-12:SB_SEMICOLON
Parsing a Block ....
2-1:KW_VAR
2-5:TK_ASSIGN(x)
2-7:SB_COLON
2-9:KW_INTEGER
2-16:SB_SEMICOLON
Parsing subroutines ....
Subroutines parsed ....
3-1:KW_SUBROUTINE
3-2:TK_ASSIGN
3-3:TK_IDENT(x)
4-6:SB_ASSIGN
Parsing an expression
4-8:SB_LPAR
Parsing an expression
4-10:ERR_INVALIDFACTOR!
```
- Bottom Status Bar:** Shows the current file is 15.kpl, line 1, column 1, spaces: 2, encoding: UTF-8, and other standard terminal options.