

BÁO CÁO THỰC HÀNH XÂY DỰNG CHƯƠNG TRÌNH DỊCH

GIẢNG VIÊN HƯỚNG DẪN: THẦY TRẦN VĨNH ĐỨC

BÁO CÁO LESSON 1: SCANNER

HỌ VÀ TÊN SINH VIÊN: NGUYỄN VĂN QUÝ

MÃ SỐ SINH VIÊN: 20210729

MÃ LỚP THỰC HÀNH: 151933

LINK GITHUB: [link](#)

PHẦN 1: HOÀN THÀNH CÁC NHIỆM VỤ CÓ TRONG SLIDES:

1. Hoàn thiện hàm *skipBlank()*

Hàm `skipBlank()` được gọi khi kí tự hiện tại có kiểu `CHAR_SPACE`:

```
Token *getToken(void)
{
    Token *token;
    int ln, cn;

    if (currentChar == EOF)
        return makeToken(TK_EOF, lineNo, colNo);

    switch (charCodes[currentChar])
    {
        case CHAR_SPACE:
            skipBlank();
            return getToken();
    }
}
```

Nếu kí tự tiếp theo vẫn là dấu cách thì thực hiện `readChar()` cho tới khi nào hết dấu cách:

```

void skipBlank()
{
    while (charCodes[currentChar] == CHAR_SPACE)
        readChar();
}

```

2. Hoàn thiện hàm skipComment()

Hàm skipComment() được gọi khi hệ thống kiểm tra thấy có 2 kí tự đứng liền nhau theo thứ tự là CHAR_LPAR, CHAR_TIMES – comment bắt đầu bởi 2 kí tự (*

```

case CHAR_LPAR:
    ln = lineNo;
    cn = colNo;
    readChar();
    if (charCodes[currentChar] == CHAR_TIMES)
    {
        readChar();
        skipComment();
        return getToken();
    }
    else if (charCodes[currentChar] == CHAR_RPAR)

```

Hàm skipComment() sẽ load lần lượt các kí tự tiếp theo tới khi hết file thì dừng hoặc tới khi 2 kí tự theo thứ tự là * và) thì sẽ dừng (kết thúc comment)

```

void skipComment()
{
    // TODO
    int ln = lineNo;
    int cn = colNo;
    while (1)
    {
        if (currentChar == EOF)
        {
            error(ERR_ENDOFCOMMENT, ln, cn);
            return;
        }

        if (charCodes[currentChar] == CHAR_TIMES)
        {
            readChar();
            if (charCodes[currentChar] == CHAR_RPAR)
            {
                readChar();
                return;
            }
        }

        if (charCodes[currentChar] == CHAR_LPAR)
        {
            ln = lineNo;
            cn = colNo;
        }
        readChar();
    }
}

```

3. Hoàn thiện hàm *readIdentKeyword*

Hàm được gọi khi gặp charCodes là CHAR_LETTER:

```

case CHAR_LETTER:
    return readIdentKeyword();

```

Nếu kí tự tiếp theo là kiểu CHAR_LETTER hoặc CHAR_DIGIT, thì thêm kí tự đó vào cuối string, sau đó kiểm tra string đó với các kí tự đã lưu bằng hàm checkKeyword, cuối cùng trả về token

```
Token *readIdentKeyword(void)
{
    // TODO
    Token *token = makeToken(TK_IDENT, lineNo, colNo);
    int count = 0;
    while (charCodes[currentChar] == CHAR_LETTER || charCodes[currentChar] == CHAR_DIGIT)
    {
        if (count <= MAX_IDENT_LEN)
        {
            token->string[count] = currentChar;
            count++;
        }
        readChar();
    }
    token->string[count] = '\0';
    TokenType type = checkKeyword(token->string);
    if (type != TK_NONE)
    {
        token->tokenType = type;
    }
    return token;
}
```

4. Hoàn thiện hàm readNumber

Hàm readNumber được gọi khi gặp charCodes là CHAR_DIGIT

```
case CHAR_DIGIT:
    return readNumber();
```

Hàm load lần lượt từ trái sang phải để có đầy đủ giá trị

```
Token *readNumber(void)
{
    Token *token = makeToken(TK_NUMBER, lineNo, colNo);
    token->value = 0;
    while ((charCodes[currentChar] == CHAR_DIGIT))
    {
        token->value = token->value * 10 + (currentChar - '0');
        readChar();
    }
    return token;
}
```

5. Hoàn thiện hàm `readConstChar`

Hàm `readConstChar` được gọi khi gặp `charCodes` là `CHAR_SINGLEQUOTE`

```
case CHAR_SINGLEQUOTE:|
    return readConstChar();
```

Hàm đọc một hằng ký tự đầu vào và trả về một token đại diện cho hằng.

```
Token *readConstChar(void)
{
    // TODO
    Token *token = makeToken(TK_CHAR, lineNo, colNo);

    readChar();
    if (currentChar == EOF)
    {
        token->tokenType = TK_NONE;
        error(ERR_INVALIDCHARCONSTANT, token->lineNo, token->colNo);
        return token;
    }

    token->string[0] = currentChar;
    token->string[1] = '\0';

    readChar();
    if (currentChar == EOF)
    {
        token->tokenType = TK_NONE;
        error(ERR_INVALIDCHARCONSTANT, token->lineNo, token->colNo);
        return token;
    }

    if (charCodes[currentChar] == CHAR_SINGLEQUOTE)
    {
        readChar();
        return token;
    }
    else
    {
        token->tokenType = TK_NONE;
        error(ERR_INVALIDCHARCONSTANT, token->lineNo, token->colNo);
        return token;
    }
}
```

6. Hoàn thiện hàm *getToken*

Dựa vào giá trị của `currentChar` trong `charCodes`, hàm `getToken` đưa ra các câu lệnh tương ứng với từng kiểu dữ liệu

```

Token *getToken(void)
{
    Token *token;
    int ln, cn;

    if (currentChar == EOF)
        return makeToken(TK_EOF, lineNo, colNo);

    switch (charCodes[currentChar])
    {
        case CHAR_SPACE:
            skipBlank();
            return getToken();
        case CHAR_LETTER:
            return readIdentKeyword();
        case CHAR_DIGIT:
            return readNumber();
        case CHAR_PLUS:
            token = makeToken(SB_PLUS, lineNo, colNo);
            readChar();
            return token;
            // ....
            // TODO
        case CHAR_MINUS:
            token = makeToken(SB_MINUS, lineNo, colNo);
            readChar();
            return token;
        case CHAR_TIMES:
            token = makeToken(SB_TIMES, lineNo, colNo);
            readChar();
            return token;
        case CHAR_SLASH:
            readChar();
            if (charCodes[currentChar] != CHAR_SLASH)
            {
                token = makeToken(SB_SLASH, lineNo, colNo-1);
                return token;
            }
    }
}

```

.....

PHẦN 2: THÊM KIỂU DỮ LIỆU STRING

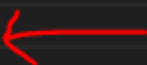
```
case KW_STRING:
    printf("KW_STRING\n");
    break;
}
```

Thêm string tại token.h

```
typedef enum {
    TK_NONE, TK_IDENT, TK_NUMBER, TK_CHAR, TK_EOF,

    KW_PROGRAM, KW_CONST, KW_TYPE, KW_VAR,
    KW_INTEGER, KW_CHAR, KW_ARRAY, KW_OF,
    KW_FUNCTION, KW_PROCEDURE,
    KW_BEGIN, KW_END, KW_CALL,
    KW_IF, KW_THEN, KW_ELSE,
    KW_WHILE, KW_DO, KW_FOR, KW_TO,

    SB_SEMICOLON, SB_COLON, SB_PERIOD, SB_COMMA,
    SB_ASSIGN, SB_EQ, SB_NEQ, SB_LT, SB_LE, SB_GT, SB_GE,
    SB_PLUS, SB_MINUS, SB_TIMES, SB_SLASH, SB_PERCENT,
    SB_LPAR, SB_RPAR, SB_LSEL, SB_RSEL,
    KW_STRING
} TokenType;
```



PHẦN 3: THÊM PHÉP TOÁN LẤY DƯ %

Thêm CHAR_PERCENT vào enum tại charcode.h

```
CHAR_COLON,
CHAR_SEMICOLON,
CHAR_PERCENT,
CHAR_SINGLEQUOTE,
CHAR_LPAR
```

Thêm lời gọi khi gặp CHAR_PERCENT:

```
case CHAR_PERCENT:
    token = makeToken(SB_PERCENT, lineNo, colNo);
    readChar();
    return token;
```

PHẦN 4: THÊM COMMENT ĐẾN CUỐI DÒNG //

Với giá trị của charCodes tại current char là CHAR_SLASH, hàm sẽ kiểm tra liệu đây là bắt đầu của phép toán chia hay đây là bắt đầu của comment kiểu C++ tới cuối dòng, từ đó gọi hàm skipSlashComment

```
case CHAR_SLASH:
    token=makeToken(SB_SLASH,lineNo,colNo)
    readChar();
    if (charCodes[currentChar] != CHAR_SLASH)
    {
        token = makeToken(SB_SLASH, lineNo, colNo-1);
        return token;
    }else{
        readChar();
        skipSlashComment();
        return getToken();
    }
```

```
void skipSlashComment()
{
    while (currentChar != '\n' && currentChar != EOF) {
        readChar();
    }
}
```

TEST CODE:

Test_1: Chạy file example1.kpl, kết quả giống nhau:

```
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> gcc -o scanner scanner.c reader.c charcode.c token.c error.c
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner ".\Test/example1.kpl">"./Test/test_result1.txt"
```

result1.txt	example1.kpl	test_result1.txt
Test > result1.txt		Test > test_result1.txt
1 1-1:KW_PROGRAM		1 1-1:KW_PROGRAM
2 1-9:TK_IDENT(Example1)		2 1-9:TK_IDENT(Example1)
3 1-17:SB_SEMICOLON		3 1-17:SB_SEMICOLON
4 2-1:KW_BEGIN		4 2-1:KW_BEGIN
5 3-1:KW_END		5 3-1:KW_END
6 3-4:SB_PERIOD		6 3-4:SB_PERIOD
7		7

Test_2: Chạy file example2.kpl, kết quả giống nhau:

```

PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> gcc -o scanner scanner.c reader.c charcode.c token.c error.c
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example1.kpl">"./Test/test_result1.txt"
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example2.kpl">"./Test/test_result2.txt"
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1>

```

Test > result2.txt	Test > test_result2.txt
1 1-1:KW_PROGRAM	1 1-1:KW_PROGRAM
2 1-9:TK_IDENT(Example2)	2 1-9:TK_IDENT(Example2)
3 1-17:SB_SEMICOLON	3 1-17:SB_SEMICOLON
4 3-1:KW_VAR	4 3-1:KW_VAR
5 3-5:TK_IDENT(n)	5 3-5:TK_IDENT(n)
6 3-7:SB_COLON	6 3-7:SB_COLON
7 3-9:KW_INTEGER	7 3-9:KW_INTEGER
8 3-16:SB_SEMICOLON	8 3-16:SB_SEMICOLON
9 5-1:KW_FUNCTION	9 5-1:KW_FUNCTION
10 5-10:TK_IDENT(F)	10 5-10:TK_IDENT(F)
11 5-11:SB_LPAR	11 5-11:SB_LPAR
12 5-12:TK_IDENT(n)	12 5-12:TK_IDENT(n)
13 5-14:SB_COLON	13 5-14:SB_COLON
14 5-16:KW_INTEGER	14 5-16:KW_INTEGER
15 5-23:SB_RPAR	15 5-23:SB_RPAR
16 5-25:SB_COLON	16 5-25:SB_COLON
17 5-27:KW_INTEGER	17 5-27:KW_INTEGER
18 5-34:SB_SEMICOLON	18 5-34:SB_SEMICOLON
19 6-3:KW_BEGIN	19 6-3:KW_BEGIN
20 7-5:KW_IF	20 7-5:KW_IF
21 7-8:TK_IDENT(n)	21 7-8:TK_IDENT(n)
22 7-10:SB_EQ	22 7-10:SB_EQ
23 7-12:TK_NUMBER(0)	23 7-12:TK_NUMBER(0)
24 7-14:KW_THEN	24 7-14:KW_THEN
25 7-19:TK_IDENT(F)	25 7-19:TK_IDENT(F)
26 7-21:SB_ASSIGN	26 7-21:SB_ASSIGN
27 7-24:TK_NUMBER(1)	27 7-24:TK_NUMBER(1)

Test_3: Chạy file example3.kpl, kết quả giống nhau:

```

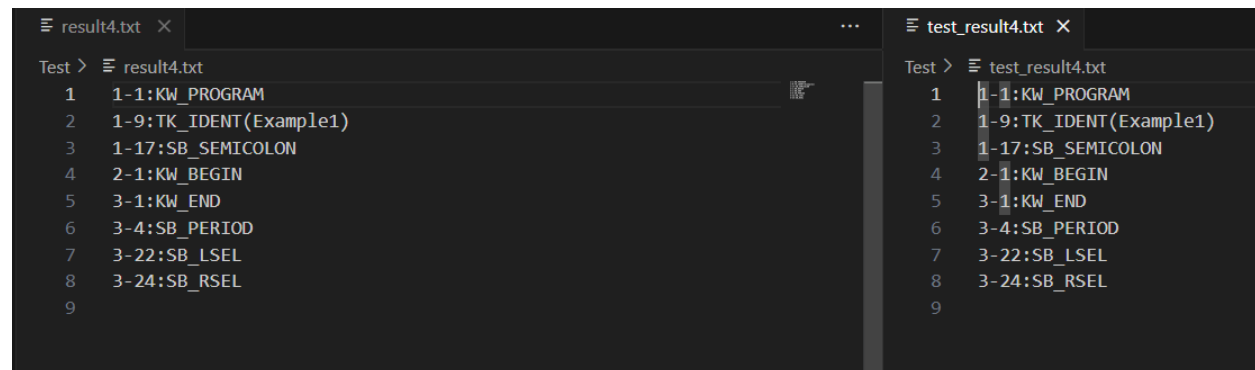
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> gcc -o scanner scanner.c reader.c charcode.c token.c error.c
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example1.kpl">"./Test/test_result1.txt"
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example2.kpl">"./Test/test_result2.txt"
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example3.kpl">"./Test/test_result3.txt"
PS D:\20241\xaydungctdich\THCTD\LESSON-1\Lesson1>

```

Test > result3.txt	Test > test_result3.txt
1 1-1:KW_PROGRAM	1 1-1:KW_PROGRAM
2 1-10:TK_IDENT(EXAMPLE3)	2 1-10:TK_IDENT(EXAMPLE3)
3 1-18:SB_SEMICOLON	3 1-18:SB_SEMICOLON
4 2-1:KW_VAR	4 2-1:KW_VAR
5 2-6:TK_IDENT(I)	5 2-6:TK_IDENT(I)
6 2-7:SB_COLON	6 2-7:SB_COLON
7 2-8:KW_INTEGER	7 2-8:KW_INTEGER
8 2-15:SB_SEMICOLON	8 2-15:SB_SEMICOLON
9 3-6:TK_IDENT(N)	9 3-6:TK_IDENT(N)
10 3-7:SB_COLON	10 3-7:SB_COLON
11 3-8:KW_INTEGER	11 3-8:KW_INTEGER
12 3-15:SB_SEMICOLON	12 3-15:SB_SEMICOLON
13 4-6:TK_IDENT(P)	13 4-6:TK_IDENT(P)
14 4-7:SB_COLON	14 4-7:SB_COLON
15 4-8:KW_INTEGER	15 4-8:KW_INTEGER
16 4-15:SB_SEMICOLON	16 4-15:SB_SEMICOLON
17 5-6:TK_IDENT(Q)	17 5-6:TK_IDENT(Q)
18 5-7:SB_COLON	18 5-7:SB_COLON
19 5-8:KW_INTEGER	19 5-8:KW_INTEGER
20 5-15:SB_SEMICOLON	20 5-15:SB_SEMICOLON
21 6-6:TK_IDENT(C)	21 6-6:TK_IDENT(C)
22 6-7:SB_COLON	22 6-7:SB_COLON
23 6-8:KW_CHAR	23 6-8:KW_CHAR
24 6-12:SB_SEMICOLON	24 6-12:SB_SEMICOLON
25 8-1:KW_PROCEDURE	25 8-1:KW_PROCEDURE
26 8-12:TK_IDENT(HANOI)	26 8-12:TK_IDENT(HANOI)

Test_4: Chạy file example4.kpl, kết quả giống nhau:

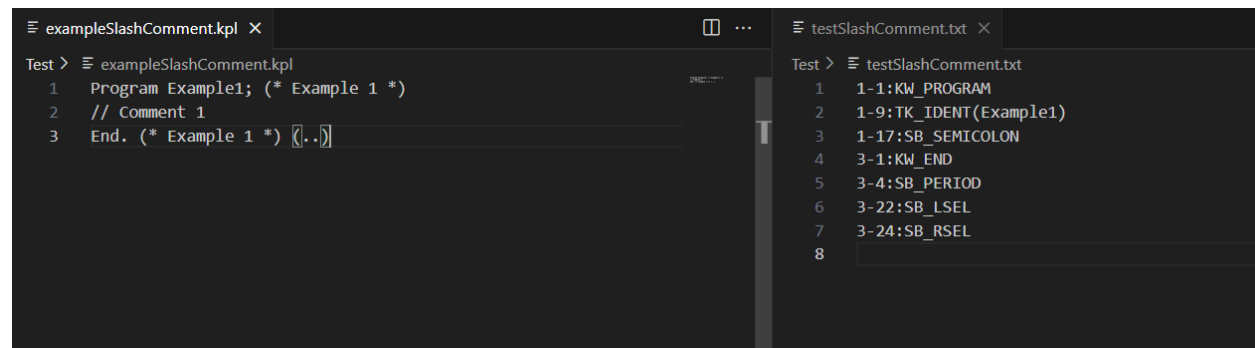
```
PS D:\20241\xyaydungctdich\THCTD\LESSON-1\Lesson1> gcc -o scanner scanner.c reader.c charcode.c token.c error.c
PS D:\20241\xyaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example1.kpl">"./Test/test_result1.txt"
PS D:\20241\xyaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example2.kpl">"./Test/test_result2.txt"
PS D:\20241\xyaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example3.kpl">"./Test/test_result3.txt"
PS D:\20241\xyaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/example4.kpl">"./Test/test_result4.txt"
```



File	Line	Token
result4.txt	1	1-1:KW_PROGRAM
	2	1-9:TK_IDENT(Example1)
	3	1-17:SB_SEMICOLON
	4	2-1:KW_BEGIN
	5	3-1:KW_END
	6	3-4:SB_PERIOD
	7	3-22:SB_LSEL
	8	3-24:SB_RSEL
	9	
test_result4.txt	1	1-1:KW_PROGRAM
	2	1-9:TK_IDENT(Example1)
	3	1-17:SB_SEMICOLON
	4	2-1:KW_BEGIN
	5	3-1:KW_END
	6	3-4:SB_PERIOD
	7	3-22:SB_LSEL
	8	3-24:SB_RSEL
	9	

Test bỏ qua comment:

```
PS D:\20241\xyaydungctdich\THCTD\LESSON-1\Lesson1> ./scanner "./Test/exampleSlashComment.kpl">"./Test/testSlashComment.txt"
PS D:\20241\xyaydungctdich\THCTD\LESSON-1\Lesson1>
```



File	Line	Token
exampleSlashComment.kpl	1	Program Example1; (* Example 1 *)
	2	// Comment 1
	3	End. (* Example 1 *)
testSlashComment.txt	1	1-1:KW_PROGRAM
	2	1-9:TK_IDENT(Example1)
	3	1-17:SB_SEMICOLON
	4	3-1:KW_END
	5	3-4:SB_PERIOD
	6	3-22:SB_LSEL
	7	3-24:SB_RSEL
	8	