

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-----\*



Môn học

Lí thuyết ngôn ngữ và phương pháp dịch

Tên đề tài

Phân tích cú pháp ngôn ngữ mini-C

Giảng viên hướng dẫn : *TS Phạm Đăng Hải*

Sinh viên thực hiện : *Phạm Minh Tâm*

Lớp : KSTN CNTT K60

Hà Nội, Ngày 31 tháng 12 năm 2018

# Mục lục

Chương 1. Giới thiệu chung về mini-C .....	3
Chương 2. Phân tích từ vựng .....	4
Chương 3. Phân tích cú pháp .....	7
Chương 4. Demo chương trình .....	11
Tài liệu tham khảo.....	14

# Chương 1. Giới thiệu chung về mini-C

Báo cáo này trình bày về cú pháp của một ngôn ngữ là tập con của ngôn ngữ C hay có thể gọi là mini-C sử dụng bộ phân tích từ vựng flex và bộ phân tích cú pháp bison. Ngôn ngữ mini-C này rất đơn giản, không bao gồm các cấu trúc phức tạp như structs, union, cấu trúc for, cấu trúc switch, ... ở ngôn ngữ C.

Ngôn ngữ cho phép sử dụng 4 kiểu giá trị gồm: void, bool, int, float.

Comment một dòng với cú pháp : `// comment`

Comment nhiều dòng : `/* comment */`

## Chương 2. Phân tích từ vựng

Từ vựng bao gồm:

- Chữ cái a-z, A-Z
- Chữ số 0-9
- Dấu đơn (){};,+-\*/%<>=![]
- Dấu kép <= >= == != || &&
- Từ khóa if else while return break new size void bool int float true false
- Tên, định danh: bắt đầu bằng chữ cái hoặc dấu gạch dưới '\_', theo sau là tổ hợp chữ cái, chữ số và dấu gạch dưới.
- Số nguyên: Tổ hợp các chữ số
- Số thực: hai tổ hợp các chữ số được ngăn cách nhau bởi dấu '.'

Ký hiệu từ tổ tương ứng

Từ vựng	Từ tổ
(){};,+-*/%<>=![]	Mã ASCII tương ứng
<= >= == !=    &&	LE GE EQ NE OR AND
if else	IF ELSE
while	WHILE
return break	RETURN BREAK
new size	NEW SIZE
void bool int float	VOID BOOL INT FLOAT
true false	TRUE FALSE
[0-9]+	INT LIT
[0-9]+.[0-9]+	FLOAT LIT
[a-zA-Z_][a-zA-Z_0-9]*	IDENT

Khai báo trong flex

```
% {  
    #include "minic.bison.h"  
    #include <stdio.h>
```

```

#include <stdlib.h>
% }

%%

"if"      return (IF);
"else"    return (ELSE);
"while"   return (WHILE);
"return"  return (RETURN);
"break"   return (BREAK);
"new"     return (NEW);
"size"    return (SIZE);

"void"    return (VOID);
"int"     return (INT);
"bool"    return (BOOL);
"float"   return (FLOAT);
"true"    return (TRUE);
"false"   return (FALSE);
"=="      return (EQ);
"!="      return (NE);
"<="      return (LE);
">="      return (GE);
"&&"      return (AND);
"||"      return (OR);
[A-Za-z_][A-Za-z0-9_]* {
    yylval.str = yytext;
    return (IDENT);
}
[0-9]+    {
    yylval.str = yytext;
    return (INT_LIT);
}
[0-9]+.[0-9]+ {
    yylval.str = yytext;
    return (FLOAT_LIT);
}
"(" return '(';
")" return ')';
"{" return '{';

```

```

"}" return '}';
 "[" return '[';
 "]" return ']';

";" return ';';
"," return ',';

"+" return '+';
 "-" return '-';
 "*" return '*';
 "/" return '/';
 "%" return '%';
 ">" return '>';
 "<" return '<';
 "=" return '=';
 "!" return '!';

"/*"([^*]\\*+[^*/])*\\**"/" ;
"//".*          ;
[ \\t\\n\\r]          ;

.      {
        printf("ky tu la");
        exit(-1);
    }

<<EOF>>          { yyterminate(); return 0; }
%%

int yywrap()
{
    return 1;
}

```

# Chương 3. Phân tích cú pháp

Cú pháp mini-C được lấy tại [1], trong đó chương trình có hai phần chính là khai báo biến toàn cục và khai báo hàm.

File bison được viết :

```
% {
    #include <stdio.h>
    #include <string.h>

    int yylex();
    void yyerror(const char *s);
% }

%union {
    int num;
    char* str;
}

%token IF ELSE WHILE RETURN BREAK NEW SIZE VOID
%token INT BOOL FLOAT
%token TRUE FALSE
%token EQ LE GE NE OR AND
%token INT_LIT
%token FLOAT_LIT
%token IDENT

%start program

%%

program
    : decl_list
    ;
decl_list
    : decl_list decl
    | decl
```

```

;
decl
    : var_decl
    | fun_decl
    ;
var_decl
    : type_spec IDENT ';'
    | type_spec IDENT '[' ']' ';';
type_spec
    : VOID | BOOL | INT | FLOAT;
fun_decl
    : type_spec IDENT '(' params ')' compound_stmt;
params
    : param_list
    | VOID;

param_list
    : param_list ',' param
    | param;
param
    : type_spec IDENT
    | type_spec IDENT '[' ']';
compound_stmt
    : '{' local_decls stmt_list '}';

local_decls
    : local_decls local_decl
    | ;

local_decl
    : type_spec IDENT ';'
    | type_spec IDENT '[' ']' ';' ;
stmt_list
    : stmt_list stmt
    | ;
stmt
    : expr_stmt
    | compound_stmt
    | if_stmt
    | while_stmt

```



```

    | return_stmt
    | break_stmt;
expr_stmt
    : expr ';'
    | ';' ;
while_stmt
    : WHILE '(' expr ')' stmt ;
if_stmt
    : IF '(' expr ')' stmt
    | IF '(' expr ')' stmt ELSE stmt ;
return_stmt
    : RETURN ';'
    | RETURN expr ';' ;
break_stmt
    : BREAK ';' ;
expr
    : IDENT '=' expr | IDENT '[' expr ']' '=' expr
    | expr OR expr
    | expr EQ expr | expr NE expr
    | expr LE expr | expr '<' expr | expr GE expr | expr '>' expr
    | expr AND expr
    | expr '+' expr | expr '-' expr
    | expr '*' expr | expr '/' expr | expr '%' expr
    | '!' expr | '-' expr | '+' expr
    | '(' expr ')'
    | IDENT | IDENT '[' expr ']' | IDENT '(' args ')' | IDENT '.' SIZE
    | INT_LIT | FLOAT_LIT | NEW type_spec '[' expr ']' ;
arg_list
    : arg_list ',' expr
    | expr ;
args
    : arg_list
    | ;

%%

```

Trong đó

- program: là ký tự bắt đầu
- decl\_list: danh sách các khai báo trong chương trình
- decl: khai báo trong chương trình, có thể là khai báo biến hoặc khai báo hàm
- var\_decl: khai báo biến toàn cục
- type\_spec: kiểu dữ liệu khai báo, là một trong các kiểu VOID, BOOL, INT, FLOAT
- fun\_decl: khai báo hàm
- params: tham số truyền vào hàm
- compound\_stmt: thân hàm
- stmt\_list: danh sách các câu lệnh bao gồm các câu lệnh if, while, được định nghĩa tương ứng bởi các từ tổ if\_stmt và while\_stmt.
- expr: biểu thức

# Chương 4. Demo chương trình

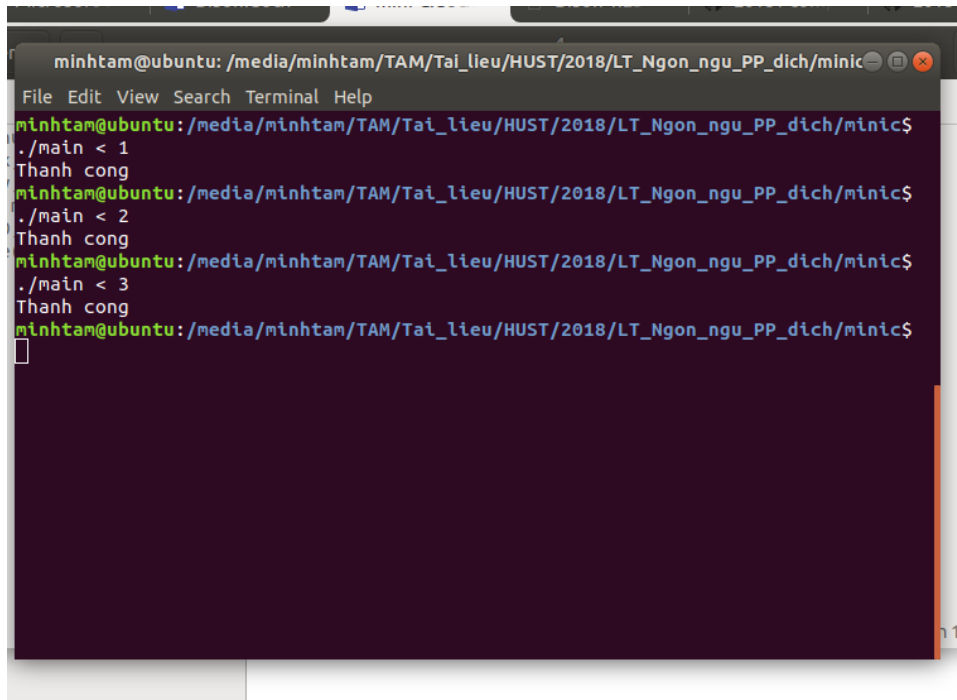
Ví dụ các chương trình sau đúng cú pháp:

```
int num;
void main(void) {
    num = iread();
    a = 1 + 2;
}
```

```
int num;
int x;
int y;
void main(void) {
    if(x==2){
        y = 1;
    }
    else{
        y = 2;
    }
}
```

```
int num;
int x;
int y;
void main(void) {
    a = 0;
    while(a>5){
        a = a+1;
        b = (a*10) + num - 2 *7 + 6/z;
        if(a >4){
            break;
        }
    }
    if(x<2){
        y = 1+2*x/y+num;
    }
}
```

```
else{  
    y = 2-x*y;  
}  
}
```



```
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$  
File Edit View Search Terminal Help  
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$  
./main < 1  
Thanh cong  
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$  
./main < 2  
Thanh cong  
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$  
./main < 3  
Thanh cong  
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$  
█
```

Ví dụ chương trình sai cú pháp do thiếu dấu ';'

```
int num;  
int x;  
int y;  
void main(void) {  
    a = 0;  
    while(a<5){  
        a = a+1  
        b = (a*10) + num - 2 *7 + 6/z;  
    }  
}
```

```
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$
File Edit View Search Terminal Help
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$
./main < 4
syntax error
minhtam@ubuntu: /media/minhtam/TAM/Tai_lieu/HUST/2018/LT_Ngon_ngu_PP_dich/minic$

```

# Tài liệu tham khảo

[1] <http://jamesvanboxtel.com/projects/minic-compiler/minic.pdf>

[2] flex & bison, John R. Levine, Tony Mason, Doug Brown , O'Reilly & Associates