205001102
Shrijith M R

Assignment 4 – Yacc

Language : python

lex.l:

```
%{
    #include<stdio.h>
    #include<stdlib.h>
    #include<string.h>
    #include "y.tab.h"
    int yylex(void);
    int yyerror(char* s);
    extern int yylval;
    int debug=0;
    int line=1;
%}

newline [\n]

/* Rules Section*/
%%
[ \t]+ ;
{newline} {++line;}

"-"?[0-9]+ {return NUM;}

"def" {return DEF;}
"for" {return FOR;}
"in" {return IN;}
"if" {return IF;}
"else" {return ELSE;}
"return" {return RETURN;}

[a-zA-Z_][a-zA-Z0-9_]* {return ID;}

(","|"("|")"|"["|"]"|":") {return *yytext;}
("+"|"-"|"*"|"/"|"//"|"%"|"**") {return ARITHMETIC_OP;}
(">"|"<"|">="|"<="|"=="|"!=") {return RELATIONAL_OP;}
("+="|"-="|"*="|"/="|"=") {return ASSIGNMENT_OP;}
"~" {return *yytext;}
"@" {return *yytext;}

[^a-zA-Z0-9_]"[".*"]" {return LIST;}
("#"|"."|","|"("|")"|"'") {return *yytext;}

. {fprintf(stderr,"Unknown token found: [%s]\n", yytext);}
```

%%

yacc.y:

```
%{
 #include <stdio.h>
 #include <stdlib.h>
    #include <fcntl.h>
    #include <unistd.h>
    int yylex(void);
    extern FILE* yyin;
    #include "y.tab.h"
    int error = 0;
    extern int line;
%}

%token DEF FOR IN IF ELSE RETURN ARITHMETIC_OP RELATIONAL_OP
ASSIGNMENT_OP ID NUM LIST

%%
long_stmt: statement long_stmt
         | statement
         ;

statement: DEF ID '(' expr ')' ':' '~' long_stmt '@'
         | FOR ID IN ID '(' expr ')' ':' '~' long_stmt '@'
         | cond_stmt
         | RETURN ID
         | assign_statement
;

cond_stmt: IF expr ':' '~' long_stmt '@' ELSE ':' '~' long_stmt
'@'
         | IF expr ':' '~' long_stmt '@'
;

assign_statement: assign_statement ASSIGNMENT_OP expr
                | expr
;

expr: expr ARITHMETIC_OP expr
    | expr RELATIONAL_OP expr
    | expr ',' expr
    | ID '(' expr ')'
    | ID
    | NUM
    | ID '[' ID ']'
    | LIST
```

```
;

%%

int yyerror(){
    fprintf(stderr, "Syntax is NOT valid!\nError at line: %d\n",
line);
    error = 1;
    return 0;
}

int yywrap(){
    return 1;
}

int main(int argc, char** argv){
    yyin = fopen(argv[1], "rt");
    if(!yyin)
    {
        printf("File not found!\n");
        return 0;
    }
    yyparse();
    if(!error){
        printf("The input program has a valid syntax!\n");
    }
    return 0;
}
```
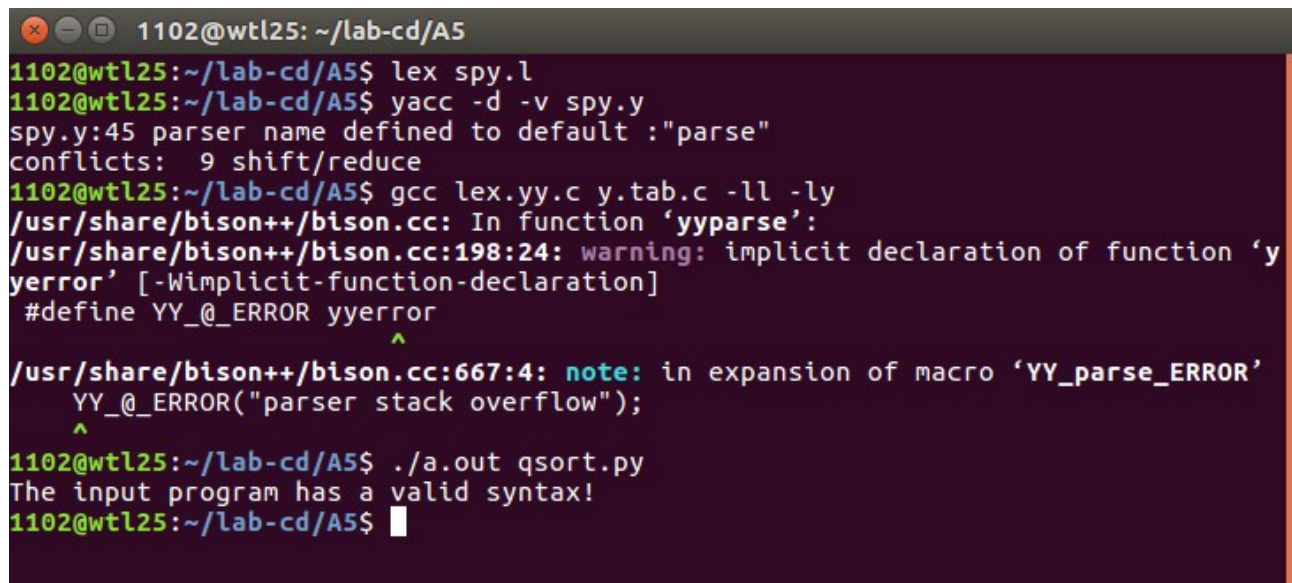
qsort.py: (input python code)

```
def partition(arr, l, r):
~
    pivot = arr[l]
    p = r
    for i in range(r, l - 1 , -1):
    ~
     if arr[i] >= pivot:
      ~
      arr[i], arr[p] = arr[p], arr[i]
      p -= 1
     @
     a = 1
    @
    p += 1
    return p
@
```

```
def quick_sort(arr, l, r):
~
    if l < r:
    ~
        p = partition(arr, l, r)
        quick_sort(arr, l, p - 1)
        quick_sort(arr, p + 1, r)
    @
@
arr = [1, 5, 4, 2, 3, 6, 8, 7]
quick_sort(arr, 0, 7)
print(arr)
```
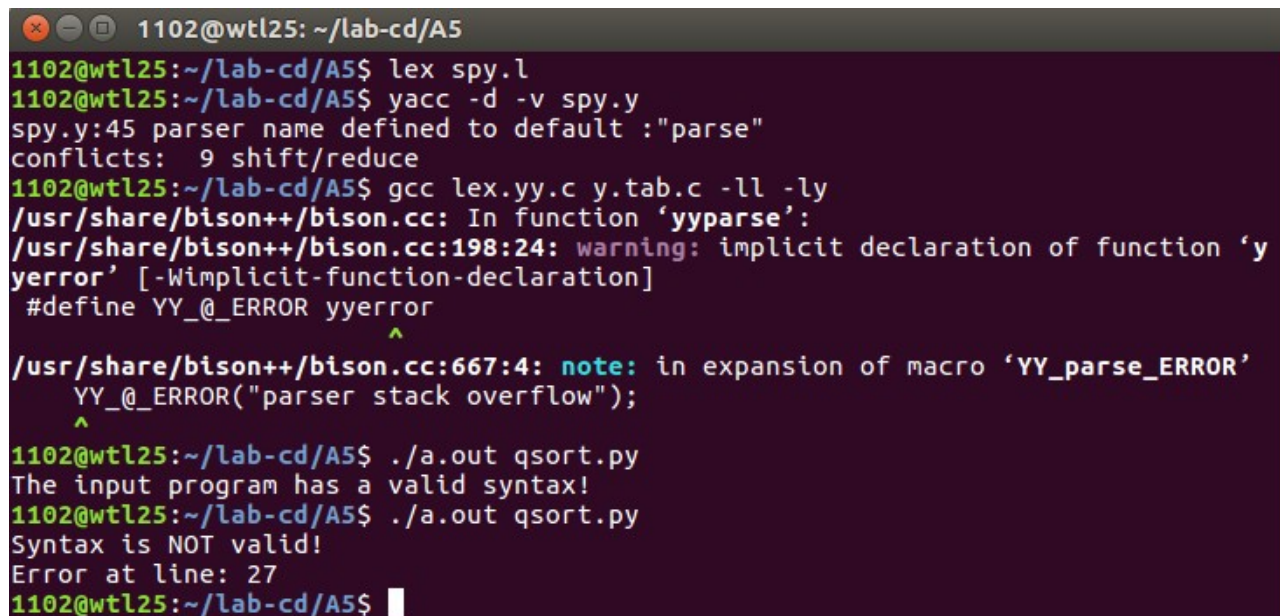
Output:

Incase of wrong syntax:

```
def partition(arr, l, r):
~
    pivot = arr[l]
    p = r
    for i in range(r, l - 1 , -1):
    ~
     if arr[i] >= pivot:
     ~
      arr[i], arr[p] = arr[p], arr[i]
      p -= 1
     @
     a = 1
    @
```

```
    p += 1
    return p
@

def quick_sort(arr, l, r):
~
    if l < r:
  ~
        p = partition(arr, l, r)
        quick_sort(arr, l, p - 1)
        quick_sort(arr, p + 1, r)
    @
@
arr = [1, 5, 4, 2, 3, 6, 8, 7 #node closing ']' - line 27
quick_sort(arr, 0, 7)
print(arr)
```

Output: