ソフトウェア演習5　レポート1

提出日　2015/10/26

13122029　高瀬正典

(1)作成したプログラムの設計情報

(1-1)全体構成

関数 init_scan:　入力ファイルが正しいかどうかを判別する。
関数 scan:　入力ファイルの文字字句解析を行う。判別した文字の判定数字を返す。ファイルの終端もしくはエラーが発生すると-1 を返す。
関数 get_linenum:　現在の行数を返す。
関数 error_scan:　char *message を入力し、そのメッセージと現在の行数を表示する。
関数 end_scan:　ファイルポインタをクローズし、終了した旨のメッセージを表示する。

main 関数でファイルが読み込まれ、init_scan 関数でそのファイルが正しく読み込まれるかの判定を行う。行数判定のための変数 linenum の初期化も行う。
その後 scan 関数でファイルの終端もしくはエラーが発生するまでファイルの文字を判別し、その判別結果を返す。返り値ことに配列 numtoken をインクリメントしてカウントし、scan 関数の終了後に結果を表示する。

(1-2)各モジュールごとの構成

配列 tokenstr:　キーワード、符号を判別するために予め予約しておく配列
配列 numtoken:　キーワード、符号をカウントするための配列
構造体 KEY:　キーワード、符号に対するカウント用の番号を与える構造体配列
配列 cbuf:　読み込んだ文字をとりあえず格納する配列
変数 num:　現在の読み込んだ文字数を示す変数
変数 end_flag:　このフラグが立つと現在読み込んでいる文字の解析が終わったことを示す変数
変数 int_flag:　このフラグが立つと現在読み込んでいる文字は数字だと解析された
変数 string_flag:　このフラグが立つと現在読み込んでいる文字はキーワードもしくは名前だと解析された
変数 aps_flag:　このフラグが立つと現在読み込んでいる文字は文字列だと解析された
変数 com_flag1:　このフラグが立つと現在読み込んでいる文字は{}で囲まれたコメントだと解析された
変数 com_flag2:　このフラグが立つと現在読み込んでいる文字は/**/で囲まれたコメントだと解析された

(1-3)各関数の外部仕様

関数 init_scan:　引数として入力ファイルを受け取り、fopen して外部変数のファイルポインタ fp に格納する。
関数 scan:　引数を取らない。外部変数の fp から 2 文字 cbuf に読み込む。2 文字目は先読みとして使用する。ファイルポインタを 1 文字戻すために fseek 関数を用いている。
その後 cbuf の判別を行う。
もし cbuf のデータが'\'であれば次の'\'が存在するまで文 字列として読み込む。
同様に cbuf のデータが'{'であれば次の'}'までをコメントとして読み込む。
cbuf のデータが'/*'であれば次の'*/'までコメントとして読み込む。
空白や改行コードであれば end_flag を立て、ループから抜ける。
改行であれば linenum をインクリメントする。
制御文字であればエラーメッセージを表示して終了する。
符号に含まれる文字なら符号として読み込み、アルファベットなら文字として読み込む。
数字であれば数字として読み込む。
その後数字であれば atoi 関数で変数 num_attr に格納し、トークンの定義文字 TNUMBER を返す。
文字列であれば TSTRING を返す。
コメントであれば 0 を返す。
キーワードもしくは NAME であれば strcpy 関数で文字列配列 string_attr に格納すし、キーワードの定義した番号もしくは TNAME を返す。
符号であれば符号の定義された番号を返す。
関数 get_linenum:　外部変数の linenum を返す。
関数 error_scan:　char *message を入力し、そのメッセージと現在の行数を表示する。
関数 end_scan:　ファイルポインタをクローズし、終了した旨のメッセージを表示する。

(2)テスト結果
(2-1)テストデータ

ファイル名:　test_white_key,sign.mpl
テストデータ:
and array begin boolean call char div do else end false if integer not of
or procedure program read readln return then true var while write writeln
+ - * = <> < <= > >= ( ) [ ] := . , : ;

ファイル名:　test_white_name,num,comment.mpl
テストデータ:
a b c d e f g h i j k l m n o p q r s t u v w x y z {mini!}
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z /*big!*/
0 1 2 3 4 5 6 7 8 9

ファイル名:　test_black_com.mpl
テストデータ:
begin
{ aaaaa
/* fffff
' ttttt
end

ファイル名:　test_black_num.mpl
テストデータ:
32766
32767
32768

ファイル名:　test_black_1024.mpl
テストデータ:
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a

(2-2)テスト結果
ファイル名:　test_white_key,sign.mpl について
実行結果を以下に示す
"program"　1
"var"　1
"array"　1
"of"　1
"begin"　1
"end"　1
"if"　1
"then"　1
"else"　1
"procedure"　1
"return"　1
"call"　1
"while"　1
"do"　1
"not"　1
"or"　1
"div"　1
"and"　1
"char"　1
"integer"　1
"boolean"　1
"readln"　1
"writeln"　1
"true"　1
"false"　1
"+"　1
"-"　1
"*"　1
"="　1
"<>"　1
"<"　1
"<="　1
">"　1
">="　1
"("　1
")"　1
"["　1
"]"　1
":="　1
"."　1
","　1
":"　1
";"　1
"read"　1
"write"　1

ファイル名:　test_white_name,num,comment.mpl について
実行結果を以下に示す
"NAME"　52
"NUMBER"　10

ファイル名:　　test_black_com.mpl について
実行結果を以下に示す
2 scan error :　not enough ' */ }
end_scan
"begin"　　1

ファイル名:　　test_black_num.mpl について
実行結果を以下に示す
3 scan error :　num error

end_scan
"NUMBER"　2

ファイル名:　　test_black_1024.mpl について
実行結果を以下に示す
"NAME"　　1100

実行時の引数が不適切な場合
File name is not specified.

(2-3)
テストデータの十分性
test_white_key,sign.mpl については、この検証で全てのキーワード、符号を網羅した。
test_white_name,num,comment.mpl については、この検証でアルファベット、数字、コメントも解析できることを示した。
以上の2つのホワイトボックステストで正常に動作する場合の各分岐命令は実証された。
test_black_com.mpl については、コメントや文字列を示す記号が片方しか無かった場合のエラーをテストした。
test_black_num.mpl については、数字が 32767 より大きい場合に出力されるエラーをテストした。
test_black_1024.mpl については、字句が 1024 文字を超えても正常に動作することが確認できた。
引数が 2 でない場合も対処できた。
以上の 4 つのブラックボックステストで境界条件のテストを行った。

(3)進捗状況
(3-1)事前計画

開始予定日	終了予定日	見積もり時間	作業内容
10/5	10/8	10	Makefile を書き、端末上でビルドして実行できる
10/8	10/12	10	vim を用いて簡単なプログラムが書けるようになる
10/12	10/14	2	配布された資料を読みなおす
10/14	10/16	5	scan 関数以外の作成
10/16	10/18	5	scan 関数の作成
10/18	10/19	1	テストプログラムの作成
10/19	10/20	2	デバッグ
10/20	10/21	5	事前計画と実際の進捗状況を書く
10/21	10/26	1	プログラムとレポートの提出

(3-3)実際の進捗状況

開始予定日	終了予定日	見積もり時間	作業内容
10/5	10/8	10	Makefile を書き、端末上でビルドして実行できる
10/8	10/16	20	vim を用いて簡単なプログラムが書けるようになる
10/17	10/17	2	配布された資料を読みなおす
10/18	10/18	8	scan 関数以外の作成
10/19	10/20	15	scan 関数の作成
10/21	10/21	1	テストプログラムの作成
10/22	10/22	4	デバッグ
10/23	10/24	5	事前計画と実際の進捗状況を書く
10/25	10/26	1	プログラムとレポートの提出

(3-4)当初の事前計画と実際の進捗との差の原因

今回のソフトウェア演習では eclipse 上でコンパイルするより毎回端末上で gcc でコンパイルして確かめたほうが良いと考えた。そこで端末上で効率良く実行するために Makefile を作成しようと考えた。また、メモ帳プログラミングはあまり行いたくので、有名な vim に手を出してみようと考えた。これが予想以上に時間がかかってしまった。

Makefile は書くことができたが、vim が手ごわかった。まずは簡単な C プログラミングからと思い、atcoder という競技プログラミングのサイトで簡単な問題を vim で数問解いた。vim の使い方について調べながら解いたのでとても時間がかかった。しかしこれで一応 vim でプログラムを書き、コンパイルはできるようになった。

その後 vim で課題のプログラムを書いていったが、まだ慣れてはいなかったので文字の入力に時間がかかった。体感的には eclipse の入力速度の 2 割程度の速さであった。

しかし課題1が終わる頃には少し慣れ、まだ eclipse ほどではないが扱えるようになった。これが事前計画と実際の進捗の差の原因である。