Document: Token Identifier 14/09/10

Aim: Prepare a C Program for Token Identifier.

Software Requirements: GNU C Compiler(GNU Compiler Collection), <stdio.h> ,<sys/types.h> ,<sys/stat.h> ,<fcntl.h>, <unistd.h> ,<string.h>

Hardware Requirements: Processor, Memory, Standard Input, Standard Output.

Description:

Program should accept the tokens as and input from the c file and process each one and should produce following.

1. It should identify the integer tokens up to max digits 5.

2. It should identify float token.

3. It should ignore white space character.

4. It should ignore comments.

5. Only + , - , / , * operators should be considered valid tokens.

6. It is assumed that the input is valid.

Algorithm:

step 1. prepare a .c c file which contains all types of tokens to get identified by the program.

step 2. give the file as an input to the main program.

step 3. Open the input file in read mode.

step 4. Read one char from file.

step 5. Identify the char type by the user-defined function isidentifier().

char isidentifier(char sym_table[][32]);

This function is called when a token starts with an alphabet.

it takes the sym_table array as an input so that it can perform the entry of the newly identified token in to it. if its a repeat token than it is ignored(for entry) and only printed with its token id number.

For identifying weather it is a repeat token or newly identified, user definded function search is called which returns an integer, i.e. if it is not a repeated token then -1 is return from search function

otherwise position number is returned.

step 6:

if step 5. fails then check weather the read char is operator from '+' '-' '*' '/' or ';'.

if it finds the match then the user-defined function named chek_2_char_operator() is called.

which checks for the occurrence of the comment as comments can also start with '/' char.

Internally, it calls the another user-defined function named check_comment() to actually check for multi line
or single line comments.

After identification of operator it finds its entry in to operator_symbol_table array.
As well as the operator with its entry number is print on to the standard output.

step 7. repeat step 4 if its not the End of File character.

Input/Output:

test.c file is supplied as and input to this program.
for testing purpose it consist of following code.

```
/*123456+10

10+11 */

123456+ 101.1

1.1 - 3.1412;
```

output:

```
Invalid integer    123456      ID0

Operator             + op0

Float Token        101.1       ID1

Float Token         1.1        ID2

Operator             - op1

Float Token        3.1412      ID3

End of line          ; op4
```