

Name – Bhargav Shamuvel Gurav

PRN – 2041009

Class – L.Y. B-Tech (Computer)

Batch – B1

Course Code – CO406U

Course Name - CDL

Practical no. 2

Aim: Write a program to identify whether a given line is a comment or not.

Theory :

If you want to identify comments in a C program, you'll need to consider C's comment syntax. In C, there are two main types of comments: single-line comments starting with `//` and multi-line comments enclosed within `/*` and `*/`. You can write a program that detects both types of comments. Here's an example in Python that identifies comments in a C program:

```
def is_single_line_comment(line):
    line = line.strip()
    return line.startswith('//')

def is_multi_line_comment_start(line):
    line = line.strip()
    return line.startswith('/*')

def is_multi_line_comment_end(line):
    line = line.strip()
    return line.endswith('*/')

def is_comment(line):
    return is_single_line_comment(line) or is_multi_line_comment_start(line)

def detect_comment_type(line, inside_multi_line_comment):
    if inside_multi_line_comment:
        if is_multi_line_comment_end(line):
            return False
        return True
    if is_single_line_comment(line) or is_multi_line_comment_start(line):
        return True
    return False

inside_multi_line_comment = False
```

```
with open("c_program.c", "r") as file:
    for line in file:
        line = line.strip()
        inside_multi_line_comment=detect_comment_type(line,
inside_multi_line_comment)
        if inside_multi_line_comment:
            print("Comment:", line)
        else:
            print("Code:", line)
```

In this Python program:

1. We define four functions: ``is_single_line_comment``, ``is_multi_line_comment_start``, ``is_multi_line_comment_end``, and ``is_comment``. These functions check if a given line is a single-line comment, the start of a multi-line comment, the end of a multi-line comment, or any type of comment, respectively.
2. The ``detect_comment_type`` function is responsible for determining whether a line is part of a multi-line comment. It checks if the line starts or ends a multi-line comment or if it is inside a multi-line comment.
3. We read the C program line by line from a file and use the ``detect_comment_type`` function to track whether we are inside a multi-line comment or not. Based on this information, we print the lines as either code or comments.

Please make sure to replace ``"c_program.c"`` with the actual path to your C program file. This program should correctly identify both single-line and multi-line comments in a C program.

Program Code:

```
#include<stdio.h>
#include<cstring>
int countSingle = 0;
int countMulti = 0;
int multiFlag = 0;
int lineCount = 0;
void commentsCheck(char line[] ,int lineCount);
int linePrint(char line[], int i)
{
    int x;
    for(x=i;x<strlen(line);x++)
    {
        printf("%c",line[x]);
    }
    return x;
}
int linePrintMulti(char line[], int i)
{
    int x=i;
    while(x<strlen(line))
    {
        if(line[x]=='*'&&line[x+1]=='/')
        {
            multiFlag = 0;
            return x;
        }
        printf("%c",line[x]);
        x++;
    }
    multiFlag = 1;
    return x;
}
int main(int argc, char* argv[])
{
    //read any text file from current directory
    char const* const fileName = "comments.c";
    FILE* file = fopen(fileName, "r");
    if(!file)
    {
```

```

printf("\n Unable to open : %s ", fileName);
return -1;
}
char line[500];
while (fgets(line, sizeof(line), file))
{
lineCount++;
commentsCheck(line , lineCount);
}
fclose(file);
printf("\nRESULT:\n");
printf("\n Single line Comments: %d ", countSingle);
printf("\n Multiple line Comments: %d ", countMulti);
return 0;
}
void commentsCheck(char line[] , int linec)
{
int i = 0;
// Single Line Comment
while(i<strlen(line) || multiFlag == 1)
{
if(line[i] == '/' && line[i+1] == '/')
{
printf("\n\nIt is a single-line comment at %d\n",linec);
countSingle++;
i = linePrint(line, i+2);
}
else if(line[i] == '/' && line[i+1] == '*')
{
printf("\n\nIt is a multi-line comment at %d\n" , linec);
multiFlag = 1;
countMulti++;
i = linePrintMulti(line,i+2);
}
else if(multiFlag==1)
{
i = linePrintMulti(line,i);
return;
}
i++;
}

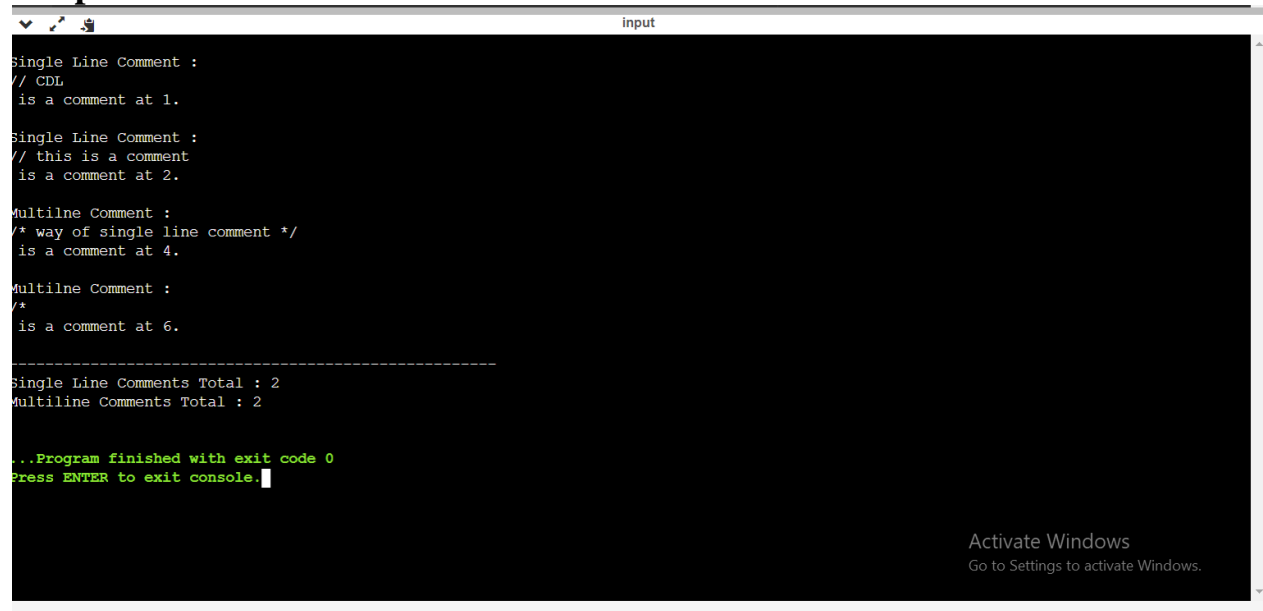
```

```
}  
return;  
}
```

Input Code (comments.c):

```
// CDL  
// this is a comment  
#include <stdio.h>  
/* way of single line comment */  
  
/*  
Multiline comment  
*/  
int main() {  
    printf("Hello, World!");  
    return 0;  
}
```

Output:



```
Input  
Single Line Comment :  
// CDL  
is a comment at 1.  
  
Single Line Comment :  
// this is a comment  
is a comment at 2.  
  
Multiline Comment :  
/* way of single line comment */  
is a comment at 4.  
  
Multiline Comment :  
/*  
is a comment at 6.  
-----  
Single Line Comments Total : 2  
Multiline Comments Total : 2  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

Conclusion : In this practical we learnt how the program identifies if there is a comment in input code.