Session 1: Scanning and Filtering a Source Program

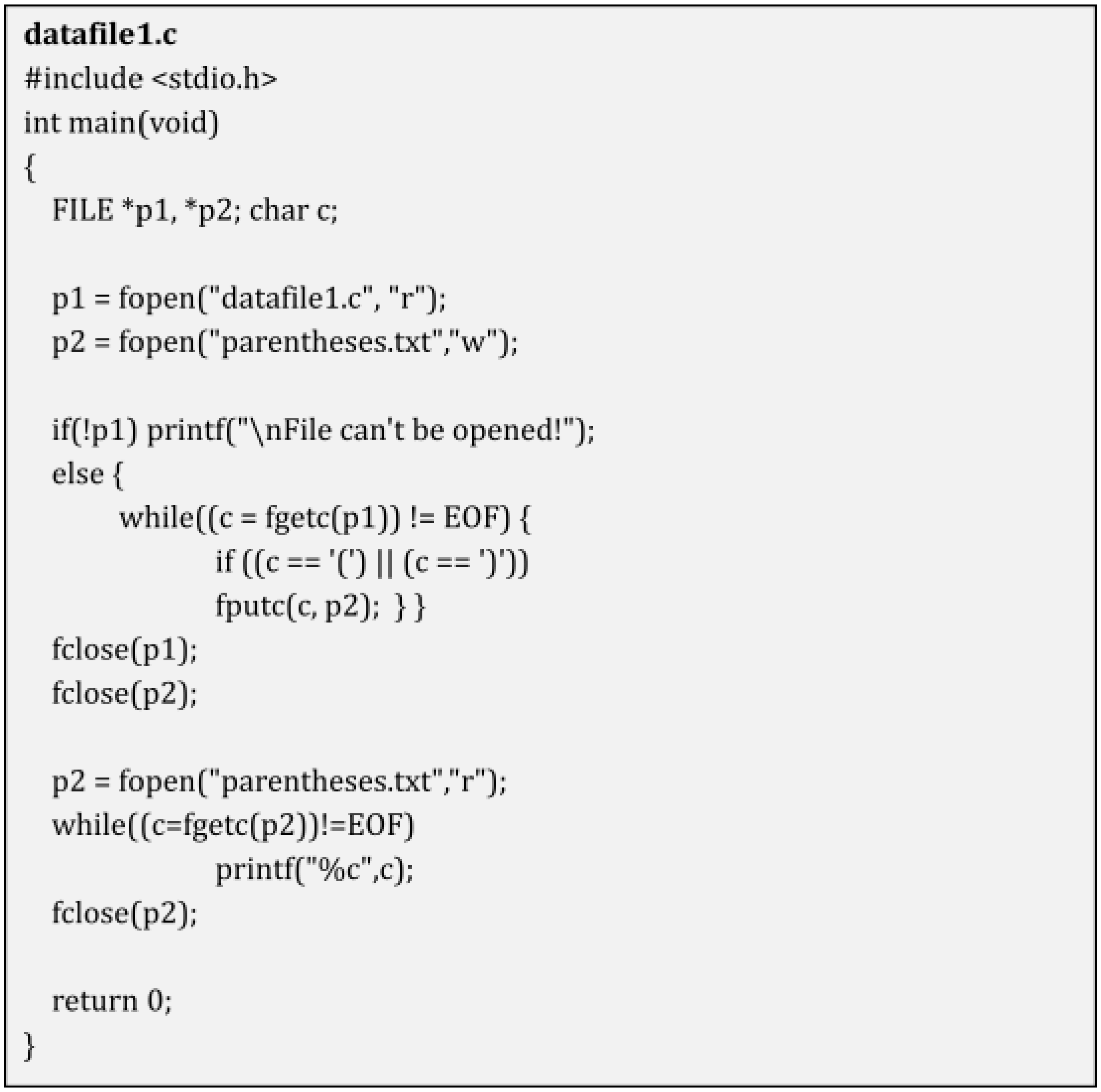
I. OBJECTIVES

To develop a program which can filter comments and newline characters from a source program.

II. DEMONSTRATION OF USEFUL RESOURCES

Extracting the sequence of occurrences of a specified character from a source program.

Sample Input: datafile1.c



datafile1.c

```
#include <stdio.h>
int main(void)
{
   FILE *p1, *p2; char c;

   p1 = fopen("datafile1.c", "r");
   p2 = fopen("parentheses.txt","w");

   if(!p1) printf("\nFile can't be opened!");
   else {
          while((c = fgetc(p1)) != EOF) {
                    if ((c == '(') || (c == ')'))
                    fputc(c, p2);  } }
   fclose(p1);
   fclose(p2);

   p2 = fopen("parentheses.txt","r");
   while((c=fgetc(p2))!=EOF)
                    printf("%c",c);
   fclose(p2);

   return 0;
}
```

Output of the program: ()()()()()((()))((()()))()()()()((()))()()

III. LAB EXERCISES

1. Write a program to print the header files used in a source program.

Sample Input: *input.c*

```
#include <stdio.h>
int main()
{
  // printf() displays the string inside quotation
printf("Hello, World!");
return 0;
}
```

Sample Output: *stdio.h*

2. Write a program to add line numbers to a source program.

Sample Input: *input.c*

Sample Output:

```
1: #include <stdio.h>
2: int main()
3: {
4:  // printf() displays the string inside quotation
5: printf("Hello, World!");
6:   return 0;
7: }
```

IV. ASSIGNMENT #1:

Step 1: Design a console application where the user will write a C program.

Step 2: Save the inserted program in a separate file.

Step 3: Filter the newly created file by removing comments and newline characters.

Step 4: Write the filtered output in another file.

Sample Input: *input1.c*

```
#include<stdio.h>

int    main(void)
{

// Single    Line Comment

printf  ("Hello");
/* Multi
     Line
                 Comment
*/
printf("World");
return 0;
}
```

Sample Output: *output.txt*

```
#include<stdio.h>int  main(void){printf (“Hello”);printf(“World”);return 0;}
```