Week2

Name : NEERAJ KUMAR
Regno. : 220905536
Roll no. : 57
Title: PRELIMINARY SCANNING APPLICATIONS

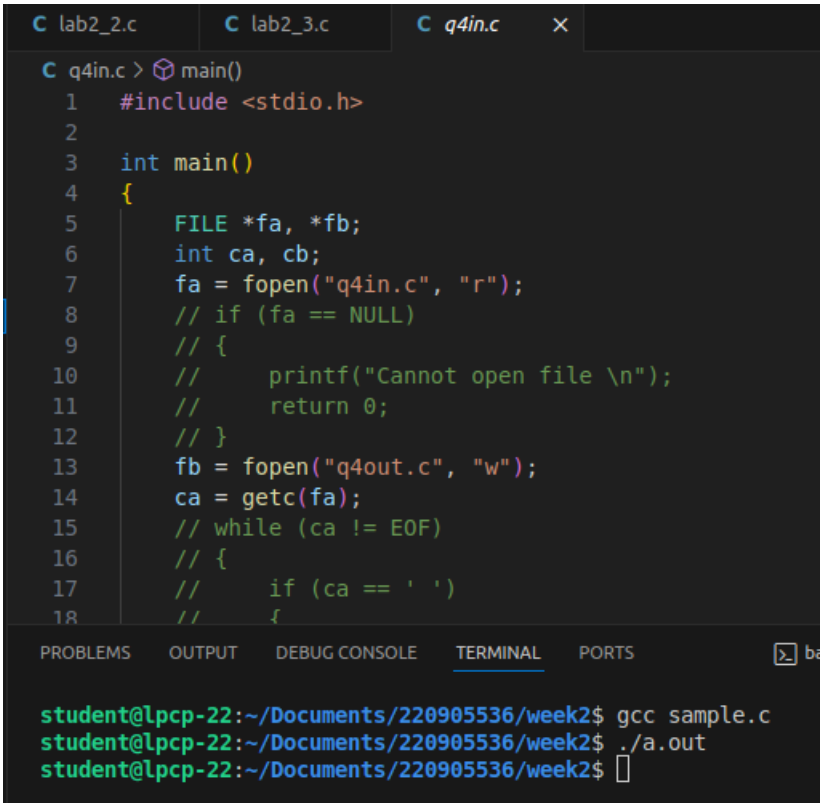
Sample Exercise:

1. **Write a C program for removal of single and multiline comments**

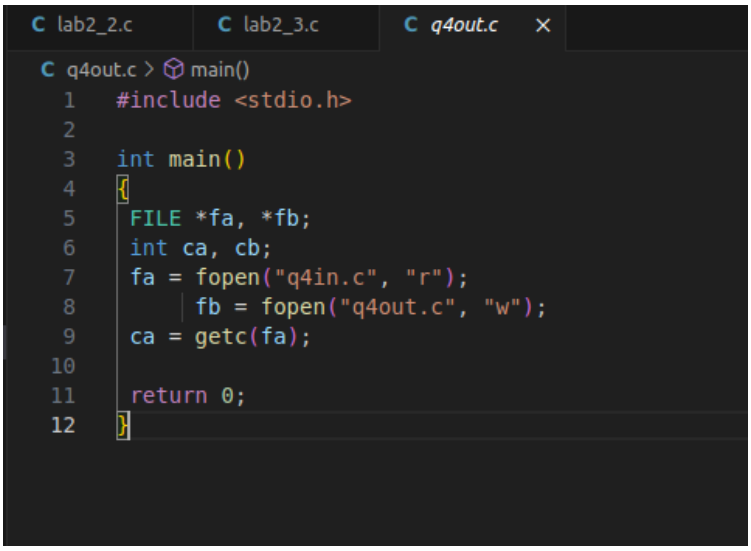
```
#include <stdio.h>

int main()
{
   FILE *fa, *fb;
   int ca, cb;
   fa = fopen("q4in.c", "r");
   if (fa == NULL)
   {
      printf("Cannot open file \n");
      return 0;
   }
   fb = fopen("q4out.c", "w");
   ca = getc(fa);
   while (ca != EOF)
   {
      if (ca == ' ')
      {
         putc(ca, fb);
         while (ca == ' ')
            ca = getc(fa);
      }
      if (ca == '/')
      {
         cb = getc(fa);
         if (cb == '/')
         {
            while (ca != '\n')
               ca = getc(fa);
         }
         else if (cb == '*')
         {
            do
            {
               while (ca != '*')
                  ca = getc(fa);
               ca = getc(fa);
            } while (ca != '/');
         }
         else
         {
```

```
            putc(ca, fb);
            putc(cb, fb);
        }
    }
    else
        putc(ca, fb);
    ca = getc(fa);
  }
  fclose(fa);
  fclose(fb);
  return 0;
}
```

Output:



After running the command the q4out.c file content where comments are removed is as follows

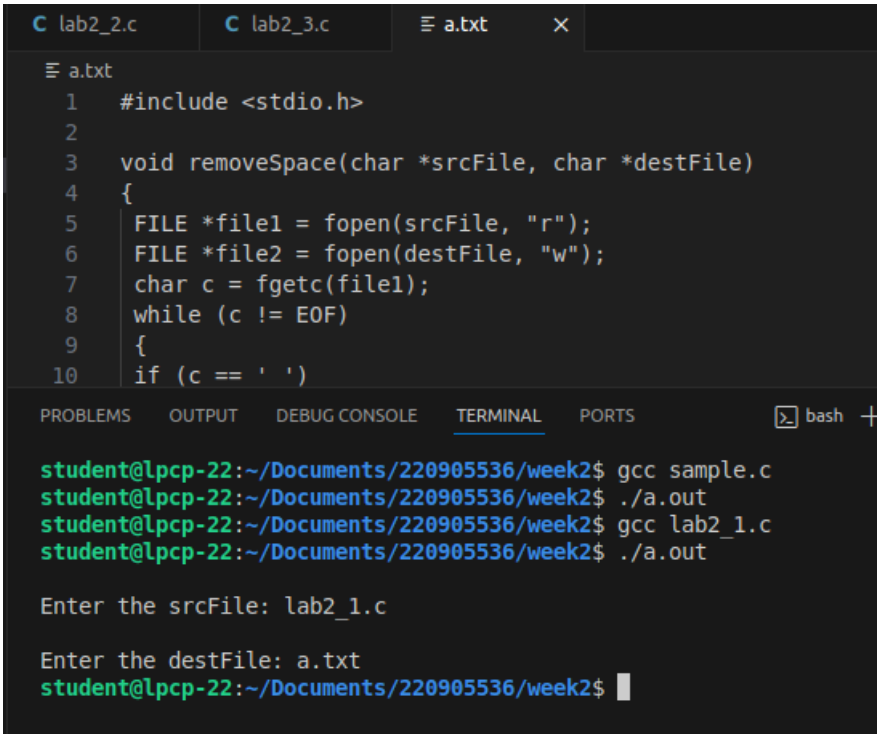


Lab Exercise:

1) **That takes a file as input and replaces blank spaces and tabs by single space and writes the output to a file**.

```
#include <stdio.h>
void removeSpace(char *srcFile, char *destFile)
{
   FILE *file1 = fopen(srcFile, "r");      //open the source file having tabs and spaces in readmode.
   FILE *file2 = fopen(destFile, "w"); //open the destination file after removing tabs and spaces.
   char c = fgetc(file1);
   while (c != EOF)
   {
      if (c == ' ')
      {
         while (c == ' ') //this loops is used to remove extra spaces and tabs
         {
            c = fgetc(file1);
         }
         if (c != ' ')
         {
            fputc(' ', file2);
            fputc(c, file2);
         }
      }
      else
      {
         fputc(c, file2);
      }
      c = fgetc(file1);
   }
   fclose(file1);
   fclose(file2);
}

int main()
{
   char srcFile[1024], destFile[1024];
   printf("\nEnter the srcFile: ");
   scanf("%s", srcFile);
   printf("\nEnter the destFile: ");
   scanf("%s", destFile);
   removeSpace(srcFile, destFile);
   return 0;
}
```

Output:

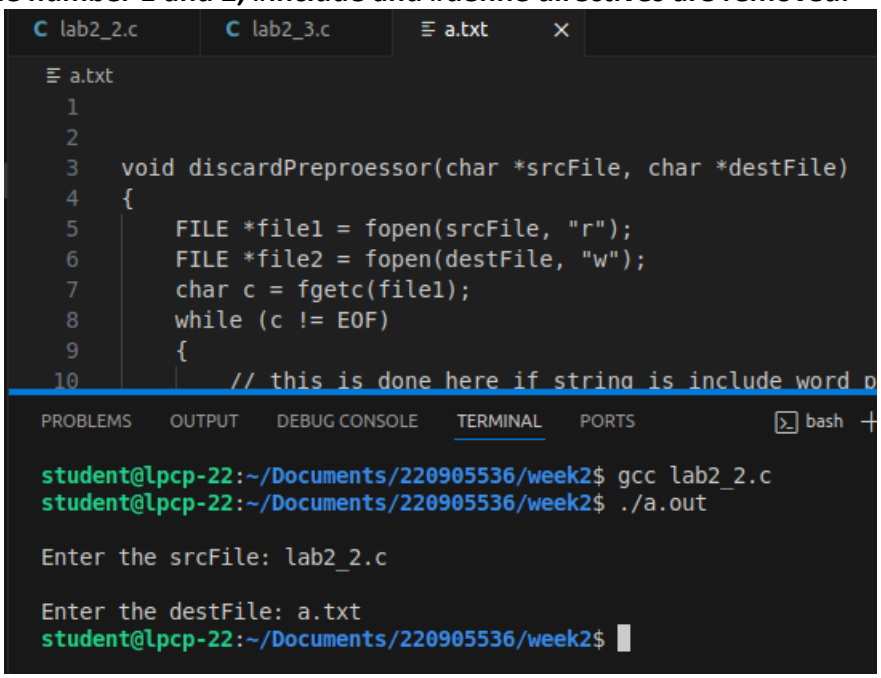


2) To discard preprocessor directives from the given input 'C' file.

```
#include <stdio.h>

#define size 1024

void discardPreproessor(char *srcFile, char *destFile)
{
   FILE *file1 = fopen(srcFile, "r");
   FILE *file2 = fopen(destFile, "w");
   char c = fgetc(file1);
   while (c != EOF)
   {
      // this is done if string has pre-processor directives word in print statement
      if (c == '\"')
      {
         fputc(c, file2);
         c = fgetc(file1);
         while (c != '\"' && c != EOF)
         {
            fputc(c, file2);
            c = fgetc(file1);
         }
         if (c == '\"')
         {
            fputc(c, file2);
         }
      }
      // logic to skip directives
      else if (c == '#')
      {
         while (c != '\n' && c != EOF)
```

```
        {
            c = fgetc(file1);
        }
    }
    // any other characters
    else
    {
        fputc(c, file2);
    }
    c = fgetc(file1);
  }
  fclose(file1);
  fclose(file2);
}

int main()
{
  char srcFile[size], destFile[size];
  printf("\nEnter the srcFile: ");
  scanf("%s", srcFile);
  printf("\nEnter the destFile: ");
  scanf("%s", destFile);
  discardPreproessor(srcFile, destFile);
  return 0;
}
```

Output: Here line number 1 and 2, #include and #define directives are removed.



3. **That takes C program as input, recognizes all the keywords and prints them in upper case.**

```
#include <stdio.h>

#include <stdio.h>
#include <string.h>
#include <ctype.h>

#define size 1024
```

```
void toUpperCase(char *word)
{
   int i = 0;
   while (word[i] != '\0')
   {
      if (word[i] >= 'a' && word[i] <= 'z')
      {
         word[i] -= 32;
      }
      i++;
   }
}

int isKeyword(char *word, char **keywords, int num_keywords)
{
   for (int i = 0; i < num_keywords; i++)
   {
      if (strcmp(word, keywords[i]) == 0)
      {
         return 1;
      }
   }
   return 0;
}

void keyword_to_Uppercase(char **keywords, int num_keywords, char *srcFile, char *destFile)
{
   char buffer[size];
   FILE *file1 = fopen(srcFile, "r");
   FILE *file2 = fopen(destFile, "w");

   if (!file1) {
      printf("Error opening files.\n");
      return;
   }

   char c = fgetc(file1);
   while (c != EOF)
   {
      int k = 0;
      // Skip non-alphabetic characters and get the next word
      while (c != EOF && !isalpha(c))
      {
         fputc(c, file2); // Write punctuation/space directly to the output file
         c = fgetc(file1);
      }

      // Read the word into the buffer
      while (c != EOF && isalpha(c))
      {
         buffer[k++] = c;
```

```
            c = fgetc(file1);
        }
        buffer[k] = '\0';

        if (k > 0 && isKeyword(buffer, keywords, num_keywords))
        {
            toUpperCase(buffer);
        }

        // Write the word (or uppercase word) to the output file
        if (k > 0)
        {
            fprintf(file2, "%s", buffer);
        }
    }

    fclose(file1);
    fclose(file2);
}

int main()
{
    char srcFile[size], destFile[size];
    char *keywords[] = {"int", "if", "void", "char","return"};
    printf("\nEnter the srcFile: ");
    scanf("%s", srcFile);
    printf("\nEnter the destFile: ");
    scanf("%s", destFile);
    int num_keywords = sizeof(keywords) / sizeof(keywords[0]);
    keyword_to_Uppercase(keywords, num_keywords, srcFile, destFile);
    return 0;
}
```

Output: Here keywords *void* became *VOID* and *char* became *CHAR* and so on.