1. **Write a C program to list all files and sub-directories in a directory.**

**Code :-**

```
#include <iostream>
#include <dirent.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>

void listFiles(const char *path, int level) {
struct dirent *entry;
DIR *dp = opendir(path);

if (!dp) {
perror("Failed to open directory");
return;
}

while ((entry = readdir(dp)) != NULL) {
// Skip the "." and ".." entries
if (strcmp(entry->d_name, ".") == 0 || strcmp(entry->d_name, "..") == 0) {
continue;
}

// Print indentation based on directory level
for (int i = 0; i < level; i++) {
std::cout << "    ";
}

std::cout << entry->d_name << std::endl;

// Construct the new path
std::string newPath = std::string(path) + "/" + entry->d_name;

// Use stat to check if the entry is a directory
struct stat statbuf;
if (stat(newPath.c_str(), &statbuf) == 0 && S_ISDIR(statbuf.st_mode)) {
listFiles(newPath.c_str(), level + 1);
}
}
if (closedir(dp) == -1) {
perror("Failed to close directory");
}
}

int main(int argc, char *argv[]) {
const char *path = "."; // Default to current directory

if (argc > 1) {
path = argv[1]; // Use path from command line arguments
}

std::cout << "Listing files in directory: " << path << std::endl;
listFiles(path, 0);

return 0;
}
```

## <u>OUTPUT :-</u>

```
Listing files in directory: .
listAllFilesAndSsub-directoriesInADirectory..cpp
listAllFilesAndSsub-directoriesInADirectory..exe
```

## 2.Write a C program to count the number of lines in a file.

## Code :-

```
#include <stdio.h>
#include <stdlib.h>

int countLines(const char *filename) {
FILE *file = fopen(filename, "r");
if (!file) {
perror("Error opening file");
return -1;
}

int count = 0;
char ch;

while ((ch = fgetc(file)) != EOF) {
if (ch == '\n') {
count++;
}
}

fclose(file);
return count;
}

int main(int argc, char *argv[]) {
if (argc < 2) {
fprintf(stderr, "Usage: %s <filename>\n", argv[0]);
return EXIT_FAILURE;
}
const char *filename = argv[1]; // Correctly assign the filename from command line
int lineCount = countLines(filename);

if (lineCount >= 0) {
printf("Number of lines in %s: %d\n", filename, lineCount);
}
return EXIT_SUCCESS;
}
```

## OUTPUT :-

```
Number of lines in filename.txt: 4
```

## 3.Write a C program to print the contents of a file.

**Code :-**

```
#include <stdio.h>
#include <stdlib.h>

void printFileContents(const char *filename) {
   FILE *file = fopen(filename, "r");
   if (!file) {
      perror("Error opening file");
      return;
   }

   char ch;
   while ((ch = fgetc(file)) != EOF) {
      putchar(ch);  // Print each character to standard output
   }

   fclose(file);
}

int main(int argc, char *argv[]) {
   if (argc < 2) {
      fprintf(stderr, "Usage: %s <filename>\n", argv[0]);
      return EXIT_FAILURE;
   }

   const char *filename = argv[1]; // Get the filename from command line arguments
   printFileContents(filename);

   return EXIT_SUCCESS;
}
```

**OUTPUT :-**

```
I Am Ankana Mondal
I come from Medinipur
I started mca degree from uem
My hobby is listening the music,travelling.
```

## 4.Write a C program to copy the contents of one file to another file.

## <u>Code :-</u>

```
#include <stdio.h>
#include <stdlib.h>

void copyFileContents(const char *source, const char *destination) {
   FILE *srcFile = fopen(source, "r");
   if (!srcFile) {
      perror("Error opening source file");
      return;
   }
   FILE *destFile = fopen(destination, "w");
   if (!destFile) {
      perror("Error opening destination file");
      fclose(srcFile);
      return;
   }
   char ch;
   while ((ch = fgetc(srcFile)) != EOF) {
      fputc(ch, destFile);  // Write each character to the destination file
   }
   fclose(srcFile);
   fclose(destFile);
   printf("Contents copied from %s to %s successfully.\n", source, destination);
}

int main(int argc, char *argv[]) {
   if (argc < 3) {
      fprintf(stderr, "Usage: %s <source_file> <destination_file>\n", argv[0]);
      return EXIT_FAILURE;
   }

   const char *source = argv[1];       // Source file
   const char *destination = argv[2];  // Destination file
   copyFileContents(source, destination);

   return EXIT_SUCCESS;
}
```

## <u>OUTPUT :-</u>

```
Contents copied from source.txt to destination.txt successfully.
```

**5.Write a C program to merge the contents of two files into a third file.**

**Code :-**

```
#include <stdio.h>
#include <stdlib.h>

void mergeFiles(const char *file1, const char *file2, const char *outputFile) {
   FILE *srcFile1 = fopen(file1, "r");
   if (!srcFile1) {
      perror("Error opening first source file");
      return;
   }

   FILE *srcFile2 = fopen(file2, "r");
   if (!srcFile2) {
      perror("Error opening second source file");
      fclose(srcFile1);
      return;
   }

   FILE *destFile = fopen(outputFile, "w");
   if (!destFile) {
      perror("Error opening destination file");
      fclose(srcFile1);
      fclose(srcFile2);
      return;
   }

   char ch;

   // Copy contents of the first file
   while ((ch = fgetc(srcFile1)) != EOF) {
      fputc(ch, destFile);
   }

   // Copy contents of the second file
   while ((ch = fgetc(srcFile2)) != EOF) {
      fputc(ch, destFile);
   }

   fclose(srcFile1);
   fclose(srcFile2);
   fclose(destFile);
```

```
    printf("Contents of %s and %s merged into %s successfully.\n", file1, file2, outputFile);
}

int main(int argc, char *argv[]) {
    if (argc < 4) {
        fprintf(stderr, "Usage: %s <file1> <file2> <output_file>\n", argv[0]);
        return EXIT_FAILURE;
    }

    const char *file1 = argv[1];        // First source file
    const char *file2 = argv[2];        // Second source file
    const char *outputFile = argv[3];   // Output file

    mergeFiles(file1, file2, outputFile);

    return EXIT_SUCCESS;
}
```

## **OUTPUT :-**

```
Contents of file1.txt and file2.txt merged into outputFile.txt successfully.
```

## 6.Write a C program to delete a file.

**Code :-**

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc < 2) {
        fprintf(stderr, "Usage: %s <filename>\n", argv[0]);
        return EXIT_FAILURE;
    }

    const char *filename = argv[1]; // Get the filename from command line arguments

    // Attempt to delete the file
    if (remove(filename) == 0) {
        printf("File %s deleted successfully.\n", filename);
    } else {
        perror("Error deleting file");
    }

    return EXIT_SUCCESS;
}
```

**OUTPUT :-**

```
File file1.txt deleted successfully.
```