1. Program to simulate
    a. Single level directory file organization technique. b. Two level directory

a) SINGLE LEVEL DIRECTORY FILE ORGANIZATION TECHNIQUE

PROGRAM

```
#include <stdio.h>
#include <string.h> // For strcmp and strcpy
#include <stdlib.h> // For exit

// Define a structure for the directory
struct
{
   char dname[10];
   char fname[10][10];
   int fcnt;
} dir;

int main() // Changed return type to int for standard compliance
{
   int i, ch;
   char f[30];

   dir.fcnt = 0;
   printf("\nEnter name of directory -- ");
   scanf("%s", dir.dname);

   while (1)
   {
      printf("\n\n1. Create File\t2. Delete File\t3. Search File \n4. Display Files\t5.
Exit\nEnter your choice -- ");
      scanf("%d", &ch);

      switch (ch)
      {
         case 1:
            if(dir.fcnt < 10) { // Check if the directory is not full
               printf("\nEnter the name of the file -- ");
               scanf("%s", dir.fname[dir.fcnt]);
               dir.fcnt++;
            } else {
               printf("Directory is full, cannot add more files.\n");
            }
            break;
         case 2:
            printf("\nEnter the name of the file -- ");
            scanf("%s", f);
            for (i = 0; i < dir.fcnt; i++)
            {
```

```
            if (strcmp(f, dir.fname[i]) == 0)
            {
               printf("File %s is deleted ", f);
               strcpy(dir.fname[i], dir.fname[dir.fcnt - 1]);
               dir.fcnt--;
               break;
            }
         }
         if (i == dir.fcnt)
            printf("File %s not found", f);
         break;
      case 3:
         printf("\nEnter the name of the file -- ");
         scanf("%s", f);
         for (i = 0; i < dir.fcnt; i++)
         {
            if (strcmp(f, dir.fname[i]) == 0)
            {
               printf("File %s is found ", f);
               break;
            }
         }
         if (i == dir.fcnt)
            printf("File %s not found", f);
         break;
      case 4:
         if (dir.fcnt == 0)
            printf("\nDirectory Empty");
         else
         {
            printf("\nThe Files are -- ");
            for (i = 0; i < dir.fcnt; i++)
               printf("\t%s", dir.fname[i]);
         }
         break;
      case 5:
         exit(0); // Correct use to exit the program
      default:
         printf("Invalid choice. Please enter a valid option.\n");
   }
  }
  // Removed getch(); as it's not standard C and not necessary here
  return 0; // Added return statement for compliance with 'int main'
}
```

OUTPUT:

Enter name of directory -- AIML
1. Create File 2. Delete File 3. Search File
1. Display Files 5. Exit Enter your choice – 1

Enter the name of the file -- A
1. Create File 2. Delete File 3. Search File
4. Display Files 5. Exit Enter your choice – 1

Enter the name of the file -- B
1. Create File 2. Delete File 3. Search File
4. Display Files 5. Exit Enter your choice – 1

Enter the name of the file -- C
1. Create File 2. Delete File 3. Search File
4. Display Files 5. Exit Enter your choice – 4

The Files are -- A B C
1. Create File 2. Delete File 3. Search File
4. Display Files 5. Exit Enter your choice – 3

Enter the name of the file – ABC File ABC not found
1. Create File 2. Delete File 3. Search File
4. Display Files 5. Exit Enter your choice – 2

Enter the name of the file – B File B is deleted
1. Create File 2. Delete File 3. Search File
4. Display Files 5. Exit Enter your choice – 5

b) TWO LEVEL DIRECTORY

PROGRAM

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

struct {
   char dname[10];
   char fname[10][10];
   int fcnt;
} dir[10];

int main() {
   int i, ch, dcnt = 0, k;
   char f[30], d[30];

   while(1) {
      printf("\n\n1. Create Directory\t2. Create File\t3. Delete File");
      printf("\n4. Search File\t\t5. Display\t6. Exit\nEnter your choice -- ");
      scanf("%d", &ch);

      switch(ch) {
         case 1:
```

```
    if(dcnt < 10) { // Check if there is room for a new directory
        printf("\nEnter name of directory -- ");
        scanf("%s", dir[dcnt].dname);
        dir[dcnt].fcnt = 0;
        dcnt++;
        printf("Directory created");
    } else {
        printf("Maximum directory limit reached.");
    }
    break;
case 2:
    printf("\nEnter name of the directory -- ");
    scanf("%s", d);
    for(i = 0; i < dcnt; i++) {
        if(strcmp(d, dir[i].dname) == 0) {
            if(dir[i].fcnt < 10) { // Check if there is room for a new file
                printf("Enter name of the file -- ");
                scanf("%s", dir[i].fname[dir[i].fcnt]);
                dir[i].fcnt++;
                printf("File created");
            } else {
                printf("Maximum file limit in directory reached.");
            }
            break; // Exit the loop once the directory is found and file is added
        }
    }
    if(i == dcnt)
        printf("Directory %s not found", d);
    break;
case 3:
    printf("\nEnter name of the directory -- ");
    scanf("%s", d);
    for(i = 0; i < dcnt; i++) {
        if(strcmp(d, dir[i].dname) == 0) {
            printf("Enter name of the file -- ");
            scanf("%s", f);
            for(k = 0; k < dir[i].fcnt; k++) {
                if(strcmp(f, dir[i].fname[k]) == 0) {
                    printf("File %s is deleted ", f);
                    dir[i].fcnt--;
                    strcpy(dir[i].fname[k], dir[i].fname[dir[i].fcnt]);
                    break; // Exit the loop once the file is found and deleted
                }
            }
            if(k == dir[i].fcnt)
                printf("File %s not found", f);
            break; // Exit the loop once the directory is found
        }
    }
    if(i == dcnt)
```

```
            printf("Directory %s not found", d);
         break;
      case 4:
         printf("\nEnter name of the directory -- ");
         scanf("%s", d);
         for(i = 0; i < dcnt; i++) {
            if(strcmp(d, dir[i].dname) == 0) {
               printf("Enter the name of the file -- ");
               scanf("%s", f);
               for(k = 0; k < dir[i].fcnt; k++) {
                  if(strcmp(f, dir[i].fname[k]) == 0) {
                     printf("File %s is found ", f);
                     break; // Exit the loop once the file is found
                  }
               }
               if(k == dir[i].fcnt)
                  printf("File %s not found", f);
               break; // Exit the loop once the directory is found
            }
         }
         if(i == dcnt)
            printf("Directory %s not found", d);
         break;
      case 5:
         if(dcnt == 0)
            printf("\nNo Directories");
         else {
            printf("\nDirectory\tFiles");
            for(i = 0; i < dcnt; i++) {
               printf("\n%s\t\t", dir[i].dname);
               for(k = 0; k < dir[i].fcnt; k++)
                  printf("\t%s", dir[i].fname[k]);
            }
         }
         break;
      default:
         exit(0);
   }
  }

  return 0; // Correct program termination with a return statement
}
```

OUTPUT:

```
1. Create Directory 2. Create File 3. Delete File
4. Search File 5. Display 6. Exit
Enter your choice -- 1
Enter name of directory -- DIR1 Directory created
1. Create Directory 2. Create File 3. Delete File
4. Search File 5. Display 6. Exit Enter your choice -- 1
```

Enter name of directory -- DIR2 Directory created
1. Create Directory 2. Create File 3. Delete File
4. Search File 5. Display 6. Exit Enter your choice -- 2
Enter name of the directory – DIR1
Enter name of the file -- A1
File created
1. Create Directory 2. Create File 3. Delete File
4. Search File 5. Display 6. Exit
Enter your choice -- 2
Enter name of the directory – DIR1
Enter name of the file -- A2
File created
1. Create Directory 2. Create File 3. Delete File
4. Search File 5. Display 6.
Exit Enter your choice – 6

Programs

1. Write a program to implement file operations(create,Delete and search) in directory structures.
2. Write a program for Directory Traversal.