1. Take multiple files as Command Line Arguments and print their inode number

```
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
main(int argc, char *argv[])
{
char d[50];
if(argc==2)
{
bzero(d,sizeof(d));
strcat(d,"ls ");
strcat(d,"-i ");
strcat(d,argv[1]);
system(d);
}
else
printf("\nInvalid No. of inputs");
}
```

Output :

```
student@ubuntu:~$ mkdir dd
student@ubuntu:~$ cd dd
student@ubuntu:~/dd$ cat >f1
hello
^z
student@ubuntu:~/dd$ cd
student@ubuntu:~$gcc -o flist.out flist.c
student@ubuntu:~$./flist.out dd
hello
46490 f1
```

2 Write a C program to find file properties such as inode number, number of hard link, File permissions, File size, File access and modification time and so on of a given file using stat() system call.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <time.h>
```

```c
void printFileProperties(struct stat stats);


int main()
{
    char path[100];
    struct stat stats;

    printf("Enter source file path: ");
    scanf("%s", path);


    // stat() returns 0 on successful operation,
    // otherwise returns -1 if unable to get file properties.
    if (stat(path, &stats) == 0)
    {
        printFileProperties(stats);
    }
    else
    {
        printf("Unable to get file properties.\n");
        printf("Please check whether '%s' file exists.\n", path);
    }

    return 0;
}


/**
 * Function to print file properties.
 */
void printFileProperties(struct stat stats)
{
    struct tm dt;

    // File permissions
    printf("\nFile access: ");
    if (stats.st_mode & R_OK)
        printf("read ");
    if (stats.st_mode & W_OK)
        printf("write ");
    if (stats.st_mode & X_OK)
        printf("execute");

    // File size
    printf("\nFile size: %d", stats.st_size);

    // Get file creation time in seconds and
    // convert seconds to date and time format
    dt = *(gmtime(&stats.st_ctime));
    printf("\nCreated on: %d-%d-%d %d:%d:%d", dt.tm_mday, dt.tm_mon, dt.tm_year +
1900,
                                              dt.tm_hour, dt.tm_min, dt.tm_sec);

    // File modification time
    dt = *(gmtime(&stats.st_mtime));
    printf("\nModified on: %d-%d-%d %d:%d:%d", dt.tm_mday, dt.tm_mon, dt.tm_year +
1900,
                                              dt.tm_hour, dt.tm_min, dt.tm_sec);
```

3 Print the type of file where file name accepted through Command Line

```
#include<stdio.h>

#include<stdlib.h>

#include<fcntl.h>

#include<unistd.h>

#include<sys/stat.h>

#include<sys/types.h>

#include<dirent.h>

int main (int argc, char *argv[])

{

struct stat fileStat;

char fnm[30];

int fd=0;

FILE *filename;

printf("Enter file name= ");

scanf("%s",fnm);

if ( ( fd = open (fnm , O_RDONLY) ) == -1){
```

```c
perror ( "open " );

system("pause");

exit (1) ;

}

if(fstat(fd, &fileStat)<0) return 1;

printf("Information for %s\n",fnm);

// expected filetype syntax here

system("pause");

return 0;

}
```

4. Write a C program to find whether a given file is present in current directory or not

```c
/**
 * C program to check whether a file exists or not.
 */

#include <stdio.h>

#include <unistd.h>

#include <io.h>

#include <sys/stat.h>

int isFileExists(const char *path);

int isFileExistsAccess(const char *path);

int isFileExistsStats(const char *path);
int main()

{

    char path[100];



    printf("Enter source file path: ");

    scanf("%s", path);





    // Check if file exists or not

    if (isFileExistsAccess(path))

    {

        printf("File exists at path '%s'\n", path);

    }

    else

    {

        printf("File does not exists at path '%s'\n", path);

    }



    return 0;

}
```

```
/**
 * Function to check whether a file exists or not.
 * It returns 1 if file exists at given path otherwise
 * returns 0.
 */
int isFileExists(const char *path)
{
   // Try to open file
   FILE *fptr = fopen(path, "r");

   // If file does not exists
   if (fptr == NULL)
      return 0;

   // File exists hence close file and return true.
   fclose(fptr);

   return 1;
}




/**
 * Function to check whether a file exists or not using
 * access() function. It returns 1 if file exists at
 * given path otherwise returns 0.
 */
int isFileExistsAccess(const char *path)
{
    // Check for file existence
    if (access(path, F_OK) == -1)
        return 0;

    return 1;
}



/**
 * Function to check whether a file exists or not using
 * stat() function. It returns 1 if file exists at
 * given path otherwise returns 0.
 */
int isFileExistsStats(const char *path)
{
    struct stat stats;

    stat(path, &stats);

    // Check for file existence
    if (stats.st_mode & F_OK)
        return 1;

    return 0;
}
```

5. Write a C program that a string as an argument and return all the files that begins with that name in the current directory. For example > ./a.out foo will return all file names that begins with foo

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

int main(void)
{
    DIR *d;
    struct dirent *dir;
    d = opendir(".");
    if (d)
    {
        while ((dir = readdir(d)) != NULL)
        {
            printf("%s\n", dir->d_name);
        }
        closedir(d);
    }
    return(0);
}
```