3. Write a C/C++ program to implement the ln/rename command using general file API’s.

```
#include <stdio.h>
#include<unistd.h>
int main(int argc, char *argv[])
{
if(argc!=3)
{
printf(“usage: %s <src_file><dest_file>\n”,argv[0]);
Return 0;
}
if(link(argv[1],argv[2])==-1)
{
printf(“link error\n”);
return 1;
}
printf(“files linked\n”);
printf(“Inode number of linked files\n”);
char str[100];
sprint(str,”ls –i %s %s \n”,argv[1],argv[2]);
system(str);
return 0;
}
```

```
Output: gcc prgm1.c
[UNIXLAB@localhost ~]$ ./a.out j.txt q.txt
Files linked
Inode number of linked files
416529 j.txt 416529 q.txt
```

Link System Call:

The link system call in Unix-like operating systems is used to create a hard link between two existing files. The hard link essentially creates a new directory entry (link) for an existing file, allowing the file to have multiple names in different locations. When a hard link is created, both the original file and the linked file share the same inode and data blocks. Changes made to one file are reflected in the other.

Syntax:

int link(const char *oldpath, const char *newpath);

oldpath: The path of the existing file.

newpath: The path for the new hard link.

Returns 0 on success and -1 on failure.

Hard Link:

Definition: A hard link is a directory entry that associates a file name with an existing inode. All hard links to a file share the same inode, and changes to one file are reflected in all hard-linked files.

Syntax (Creating):

```
link("existing_file", "new_hard_link");
```

Soft Link (Symbolic Link):

Definition: A symbolic link, or soft link, is a special type of file that contains a reference to another file or directory in the form of a pathname. It acts as a pointer to the target file or directory.

Syntax (Creating):

```
symlink("target_file", "new_soft_link");
```

Inode Number:

Definition: In Unix-like file systems, an inode is a data structure on disk that stores information about a file or a directory, such as its size, permissions, timestamps, and pointers to data blocks.

Purpose: The inode number is a unique identifier assigned to each inode in a file system. It allows the operating system to locate and manage files efficiently.

Accessing Inode Number: You can access the inode number of a file using the ls -i command or programmatically using system calls.

system(str) call is used to run a shell command that lists the Inode numbers of the linked files, providing additional information about the linked files in the output.

7.Write C/C++ program to read and display the last 10 character’s of the input file.

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
FILE *fp;
char ch;
int num;
long length;
printf("Enter the value of num:");
scanf("%d",&num);
fp=fopen("file.txt","r");
if(fp==NULL)
{
puts("Cannot open this file");
exit(1);
}
fseek(fp,-1,SEEK_END);
length=ftell(fp);
fseek(fp,(length-num),SEEK_SET);
do
{
ch=fgetc(fp);
putchar(ch);
}while(ch!=EOF);
fclose(fp);
return(0);
}
```

Output

```
cat > l.txt
hello world hai beautiful girl

gcc p55.c
[UNIXLAB@localhost ~]$ ./a.out p55.c
Enter the value of num:4
Girl
```

6.Write a C/C++ program to query and display the different attributes associated with a file.

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

int main(int argc, char *argv[])
{
struct stat sb;
if(argc!=2)
{
fprintf(stderr,"usage: %s <pathname>\n", argv[0]);
exit(EXIT_FAILURE);
}
if(stat(argv[1],&sb)==-1)
{
perror("stat");
exit(EXIT_FAILURE);
}

printf("file type:          ");
switch(sb.st_mode & S_IFMT)
{
case S_IFBLK: printf("block device file\n");
           break;
case S_IFCHR: printf("character device file\n");
           break;
case S_IFDIR: printf("directory\n");
           break;
case S_IFIFO: printf("FIFO/pipe\n");
           break;
case S_IFLNK: printf("symlink\n");
           break;
case S_IFREG: printf("regular file\n");
           break;
case S_IFSOCK: printf("socket\n");
            break;
default:       printf("regular file\n");
            break;
}
printf("Inode number:   %ld\n", (long) sb.st_ino);
printf("Mode:   %lo(octal)\n", (unsigned long) sb.st_mode);
printf("Blocks allocated:   %lld\n", (long long) sb.st_blocks);
exit(EXIT_SUCCESS);
}
```

Output: gcc prgm.c

```
[UNIXLAB@localhost ~]$ ./a.out  prgm.c

File type: regular

Inode number: 1067168

Mode: 100664(octal)

Blocks allocated: 8
```