```
/* File Copy   Operation *

#include<fcntl.h>
#include<sys/stat.h>
 #define BUFSIZE 1024
 int main(void)
{
    int fd1,fd2,n;
    char buf[BUFSIZ];
    fd1=open("/etc/passwd", O_RDONLY);
     fd2=open("passwd.bak",O_WRONLY |O_CREAT|O_TRUNC,
S_IRUSR|S_IWUSR|S_IRGRP| S_IWGRP|S_IROTH);   /*Mode 664  */
 while((n=read(fd1,buf,BUFSIZE))>0)
       write(fd2,buf,n);
 close(fd1);
close(fd2);
 exit(0);
}
```

```
/* Reverse Reading a FIle */
#include<fcntl.h>
#include<unistd.h>
int main(int argc, char ** argv) {
Char buf; int size,fd;
Fd=open(argv[1],O_RDONLY);
Size=lseek(fd,-1,SEEK_END);      /*Pointer taken to EOF -1 …..*/
 while(size -- >=0) {
read(fd,&buf,1);                                        /*  Read 1
char  at a time  */
 write(STDOUT_FILENO,&buf,1);           /* And write it immediately
*/
 lseek(fd, -2,SEEK_CUR);                     /*Now move File pointer
back by 2 char */
}}
```

/Write a shell script to find and display all the links to a file specified as the first argument to the script.

```
#!/bin/sh
if [ $# -ne 2 ]
then
echo "Error in number of parameters"
exit 1
fi
dirpath = $2
inode = `ls – aliR | grep "$1" | cut –c  1 -7`
ls –aliR $dirpath | grep "$inode"
```

Output:

```
$sh link.sh f7 sun

1790546 –rw-rw-r- - 5 4aut49 4aut49 11 feb26 08:09 f7
1790546 –rw-rw-r- - 5 4aut49 4aut49 11 feb26 08:09 f7
1790546 –rw-rw-r- - 5 4aut49 4aut49 11 feb26 08:09 f7
```

Write a shell script that accepts 2 file names as arguments sorts both to temporary files merges the sorted files to the standard output and finally deletes the temporary files.

```
#!/bin/sh
if [ $# -ne 2 ]
then
echo “Error in number of arguments”
exit 1
fi
sort $1 > temp
sort $2 > temp1
sort –m temp temp1
rm temp temp1
```

Output:

```
$sh sort.sh temp temp1

Abhay
Barthy
Dinesh
Jai shree
Kishore
Oasis
Suresh
```

Write a shell script that accepts valid log-in names as arguments and print their corresponding home directories.

```
#!/bin/sh
if [ $# -eq 0 ]
then
echo “ERROR!! No parameters”
exit 1;
fi
for i in $*
do
echo “The home directory of $i is”
grep $i /etc/passwd | cut –d “:” –f  6
done
```

Output:

```
$sh home.sh 4aut44

The home directory of 4aut44 is
/home/4aut44
```

Write a C program that accepts valid file names as command line arguments and for each of the arguments, prints the type of the file (Regular file, directory file, Character special file, Block special file, symbolic link etc.,)

```
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>
#include<sys/stat.h>
#include<sys/types.h>
int main(int argc,char *argv[])
{
        struct stat statv;
        int i;
        for(i=0;i<argc;i++)
                {
                if(lstat(argv[i],&statv)==-1)
                        {
                        printf("%s is invalid file\n",argv[i]);
                        continue;
                        }
                if(S_ISDIR(statv.st_mode))
                        printf("%s is a directory file\n",argv[i]);
                else if(S_ISREG(statv.st_mode))
                        printf("%s is a regular file\n",argv[i]);
                //else if(S_ISBLHelse if(S_ISREG(statv.st_mode))
                  //      printf("%s is a regular file\n",argv[i]);
                else if(S_ISBLK(statv.st_mode))
                        printf("%s is a block device file\n",argv[i]);
                else if(S_ISCHR(statv.st_mode))
                        printf("%s is a character device 
file\n",argv[i]);
                else if(S_ISFIFO(statv.st_mode))
                        printf("%s is a fifo file\n",argv[i]);
                else if(S_ISLNK(statv.st_mode))
                        printf("%s is a symbolic link 
file\n",argv[i]);
                }
return 0;
}
```

```
/*Write a C program that accepts valid file names as command line 
arguments and for each of the arguments, prints the type of the file 
(Regular file, directory file, Character special file,  Block special 
file, symbolic link etc.,)*/
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>
#include<sys/stat.h>
#include<sys/types.h>
int main(int argc,char *argv[])
{
        struct stat statv;
        int i;
        for(i=0;i<argc;i++)
```

```
                {
                if(lstat(argv[i],&statv)==-1)
                        {
                        printf("%s is invalid file\n",argv[i]);
                        continue;
                        }
                if(S_ISDIR(statv.st_mode))
                        printf("%s is a directory file\n",argv[i]);
                else if(S_ISREG(statv.st_mode))
                        printf("%s is a regular file\n",argv[i]);
                //else if(S_ISBLHelse if(S_ISREG(statv.st_mode))
                  //      printf("%s is a regular file\n",argv[i]);
                else if(S_ISBLK(statv.st_mode))
                        printf("%s is a block device file\n",argv[i]);
                else if(S_ISCHR(statv.st_mode))
                        printf("%s is a character device 
file\n",argv[i]);
                else if(S_ISFIFO(statv.st_mode))
                        printf("%s is a fifo file\n",argv[i]);
                else if(S_ISLNK(statv.st_mode))
                        printf("%s is a symbolic link 
file\n",argv[i]);
                }
return 0;
}
```