Program 19: Write a c program to emulate the unix ln command

```
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
#include<string.h>
int main(int argc, char *argv[])
{
if(argc < 3 || argc > 4 || (argc == 4 && strcmp(argv[1],"-s")))
{
printf("Usage: ./a.out [-s] <org_file> <new_link>\n");
return 1;
}
if(argc == 4)
{
if((symlink(argv[2],argv[3])) == -1)
printf("Cannot create symbolic link\n") ;
else
printf("Symbolic link created\n") ;
}
else
{
if((link(argv[1], argv[2])) == -1)
printf("Cannot create hard link\n") ;
else
printf("Hard link created\n") ;
}
return 0;
}
```

Output:

```
bmsce@bmsce-Precision-T1700:~$ ./prog19out
        Usage: ./a.out [-s] <org_file> <new_link>
bmsce@bmsce-Precision-T1700:~$ ./prog19out hello.c hello1
        Hard link created
bmsce@bmsce-Precision-T1700:~$ ./prog19out -s hello.c hello2
        Symbolic link created
```

Program 20: Write a c program which demonstrates interprocess communication between the reader Process and a writer process

```
#include<sys/types.h>
#include<unistd.h>
#include<fcntl.h>
#include<sys/stat.h>
#include<string.h>
#include<errno.h>
#include<stdio.h>
int main(int argc, char* argv[])
{
int fd;
char buf[256];
if(argc != 2 && argc != 3)
{
printf("USAGE %s <file> [<arg>]\n",argv[0]);
return 0;
}
mkfifo(argv[1],S_IFIFO | S_IRWXU | S_IRWXG | S_IRWXO );
if(argc == 2)
{
fd = open(argv[1], O_RDONLY|O_NONBLOCK);
while(read(fd, buf, sizeof(buf)) > 0)
printf("%s",buf);
}
else
{
fd = open(argv[1], O_WRONLY);
write(fd,argv[2],strlen(argv[2]));
}
close(fd);
}
```

Output:

```
bmsce@bmsce-Precision-T1700:~$ ./progm20out abc
        hello welcome to unix
        get started
�bmsce@bmsce-Precision-T1700:~$ ./progm20out abc "Have a good day"
bmsce@bmsce-Precision-T1700:~$ ./progm20out abc
        Have a good dayo unix
        get started
```