Lab- 12

1. Program to emulate 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 a 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;
}
```

```
gcc lncommand.c
./a.out 1 2
3 4
$./a.out 1.c
2
./a.out -s 1 a.c 22
Symbolic link created
```

2 Program that prints POSIX defined configuration options supported on any system using feature test macros

```
#define _POSIX_SOURCE
#define _POSIX_C_SOURCE 199309L
#include<stdio.h>
#include<unistd.h>
int main(){
    #ifdef _POSIX_JOB_CONTROL
    printf("System supports job control\n");
    #else
    printf("System does not support job control\n");
    #endif
    #ifdef _POSIX_SAVED_IDS
    printf("System supports saved set UID & saved set GID");
    #else
    printf("System does not support saved set UID & saved set GID");
    #endif
    #ifdef
    _POSIX_CHOWN_RESTRICTED
    printf("chown_restricted option is %d\n", _POSIX_CHOWN_
                                        RESTRICTED);
    #else
    printf("System does not support chown_restricted option");
    #endif
    #ifdef
    _POSIX_NO_TRUNC
    printf("Pathname trunc option is %d\n", POSIX_NO_TRUNC);
    #else
    printf("System does not support system wide pathname
            trunc option\n");
    #endif
    #ifdef
    _POSIX_VDISABLE
    printf("Disable character for terminal files is %d\n",
                                        POSIX_VDISABLE);
    #else
    printf("System does not support _POSIX_VDISABLE\n");
```

```
#endif
return 0;
}
```

O/P:

System supports job control
System supports set-UID & set-GID
chown_restricted
option is 1
Pathname
trunc option is 1
Disable
Character for terminal files is 0

3. Program which demonstrates interprocess communication b/w reader process & writer process.
Use mkfifo, open, read, write, close API's

```
#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);
}
```

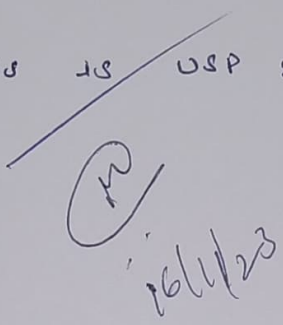
O/P:

Terminal 1

```
gcc third.c
./a.out fifo1
"this is OSP Exam; CD bbb"
```

Terminal 2

```
nano third.c
./a.out fifo1
this is OSP Exam; CD bmf
```



16/11/23