P1.c

1 #include<stdio.h>

2 #include<fcntl.h>

3 #include<string.h>

4 void\* thread1(void \*p)

5 {

6 int fd;

7 char a[20];

8 fd=open("np1",O\_WRONLY);

9 while(1)

10 {

11 scanf("%s",a);

12 write(fd,a,strlen(a)+1);

13 }

14 }

15 void\* thread2(void \*p)

16 {

17 int fd;

18 char b[20];

19 fd=open("np2",O\_RDONLY);

20 while(1)

21 {

22 read(fd,b,sizeof(b));

23 printf("%s\n",b);

24 }

25 }

26

27 main()

28 {

29 pthread\_t t1,t2;

30 mkfifo("np1",0664);

31 mkfifo("np2",0664);

32 pthread\_create(&t1,0,thread1,0);

33 pthread\_create(&t2,0,thread2,0);

34 while(1);

35 }

P2.c

1 #include<stdio.h>

2 #include<fcntl.h>

3 #include<string.h>

4 void\* thread1(void \*p)

5 {

6 int fd;

7 char a[20];

8 fd=open("np1",O\_RDONLY);

9 while(1)

10 {

11 read(fd,a,sizeof(a));

12 printf("%s\n",a);

13 }

14 }

15 void\* thread2(void \*p)

16 {

17 int fd;

18 char b[20];

19 fd=open("np2",O\_WRONLY);

20 while(1)

21 {

22 scanf("%s",b);

23 write(fd,b,strlen(b)+1);

24 }

25 }

26

27 main()

28 {

29 pthread\_t t1,t2;

30 mkfifo("np1",0664);

31 mkfifo("np2",0664);

32 pthread\_create(&t1,0,thread1,0);

33 pthread\_create(&t2,0,thread2,0);

34 while(1);

35 }

P3.c

1 //common data access b/w threads

2 //it is easy that data sharing b/w threads no need of any ipc

3 //at the same time when there is a common data b/w threads it is required synchrinization to

4 //access the data .

5 #include<stdio.h>

6 #include<pthread.h>

7 int g=0;

8 void\* thread1(void \*p)

9 {

10 printf("in thread1->g:%d\n",g);

11 g++;

12 printf("in thread1->g:%d\n",g);

13 sleep(5);

14 printf("in thread1->g:%d\n",g);

15 printf("thread1 terminated...\n");

16 }

17 void\* thread2(void \*p)

18 {

19 sleep(2);

20 printf("in thread2:%d\n",g);

21 g=100;

22 printf("in thread2 g value updated...\n");

23 sleep(2);

24 printf("thread2 terminated...\n");

25 }

26 main()

27 {

28 pthread\_t t1,t2;

29 pthread\_create(&t1,0,thread1,0);

30 pthread\_create(&t2,0,thread2,0);

31 while(1);

32 }