**Lab 5: IPC-1: PIPE, FIFO**

**Lab Exercises:**

1. Write a producer and consumer program in C using the FIFO queue. The producer should write a set of 4 integers into the FIFO queue and the consumer should display the 4 integers.

**Producer**

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<unistd.h>**

**#include<sys/types.h>**

**#include<limits.h>**

**#include<fcntl.h>**

**#include<sys/msg.h>**

**#include<sys/stat.h>**

**#include<string.h>**

**#define FIFO\_NAME "my\_fifo"**

**#define BUFFER\_SIZE 1000**

**int main(int argc, char \*argv[])**

**{**

**int pipe\_fd;**

**int res;**

**int open\_mode=O\_WRONLY;**

**int n=0;**

**char buffer[BUFFER\_SIZE+1];**

**if(access(FIFO\_NAME,F\_OK)==-1)**

**{**

**res=mkfifo(FIFO\_NAME,0777);**

**if(res!=0)**

**{**

**fprintf(stderr, "Could not create file%s\n",FIFO\_NAME );**

**exit(EXIT\_FAILURE);**

**}**

**}**

**printf("Process %d opening FIFO O\_WRONLY\n",getpid());**

**pipe\_fd=open(FIFO\_NAME,open\_mode);**

**printf("Process %d result %d\n",getpid(),pipe\_fd);**

**if (pipe\_fd!=-1)**

**{**

**printf("Enter 4 numbers\n");**

**while(n<4)**

**{**

**scanf("%s",buffer);**

**res=write(pipe\_fd,buffer,BUFFER\_SIZE);**

**if(res==-1)**

**{**

**fprintf(stderr, "Write Error on Pipe\n");**

**exit(EXIT\_FAILURE);**

**}**

**n++;**

**}**

**(void)close(pipe\_fd);**

**}**

**else**

**exit(EXIT\_FAILURE);**

**printf("Process %d Finished\n",getpid() );**

**exit(EXIT\_SUCCESS);**

**}**

**Consumer**

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<unistd.h>**

**#include<sys/types.h>**

**#include<limits.h>**

**#include<fcntl.h>**

**#include<sys/msg.h>**

**#include<sys/stat.h>**

**#include<string.h>**

**#define FIFO\_NAME "my\_fifo"**

**#define BUFFER\_SIZE 1000**

**int main(int argc, char \*argv[])**

**{**

**int pipe\_fd;**

**int res;**

**int open\_mode=O\_RDONLY;**

**int n=0;**

**char buffer[BUFFER\_SIZE+1];**

**memset(buffer,'\0',sizeof(buffer));**

**printf("Process %d opening FIFO O\_RDONLY\n",getpid());**

**pipe\_fd=open(FIFO\_NAME,open\_mode);**

**printf("Process %d result %d\n",getpid(),pipe\_fd);**

**if (pipe\_fd!=-1)**

**{**

**do**

**{**

**res=read(pipe\_fd,buffer,BUFFER\_SIZE);**

**printf("%s\n",buffer );**

**n++;**

**}while(n<4);**

**(void)close(pipe\_fd);**

**}**

**else**

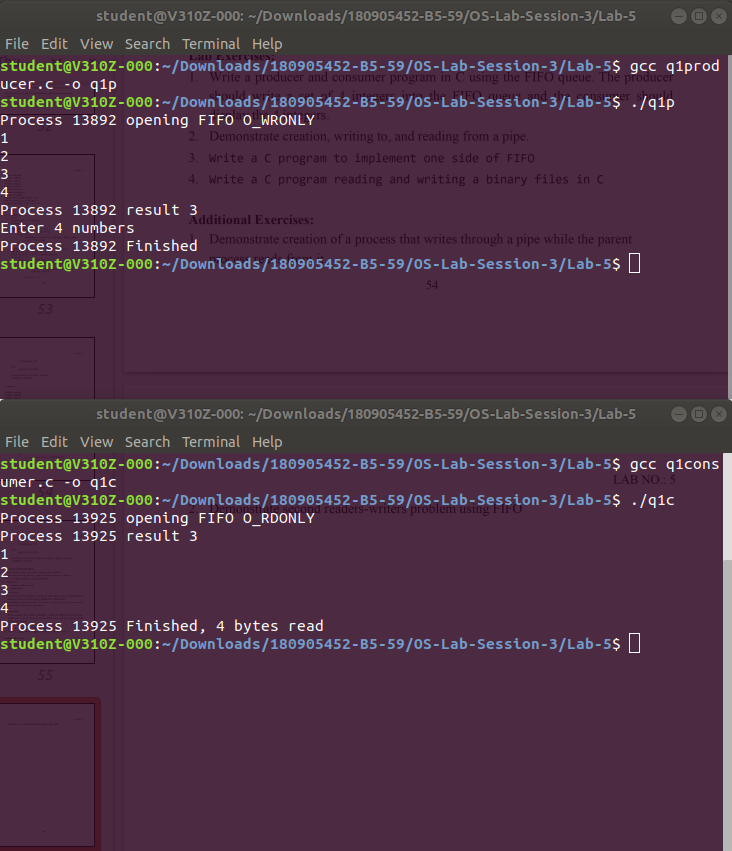
**exit(EXIT\_FAILURE);**

**printf("Process %d Finished, %d bytes read\n",getpid(),n );**

**exit(EXIT\_SUCCESS);**

**}**

**Output**



1. Demonstrate creation, writing to, and reading from a pipe.

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<sys/ipc.h>

#include<sys/msg.h>

#include<string.h>

int main(int argc, char \*argv[])

{

int n;

int fd[2];

char buf[1025];

char \*data="Mid Term Lab Exams from tomorrrow!";

pipe(fd);

write(fd[1],data,strlen(data));

if(n=read(fd[0],buf,1024)>=0)

{

buf[n]=0;

printf("Read %d bytes from pipe\"%s\"\n",n,buf);

}

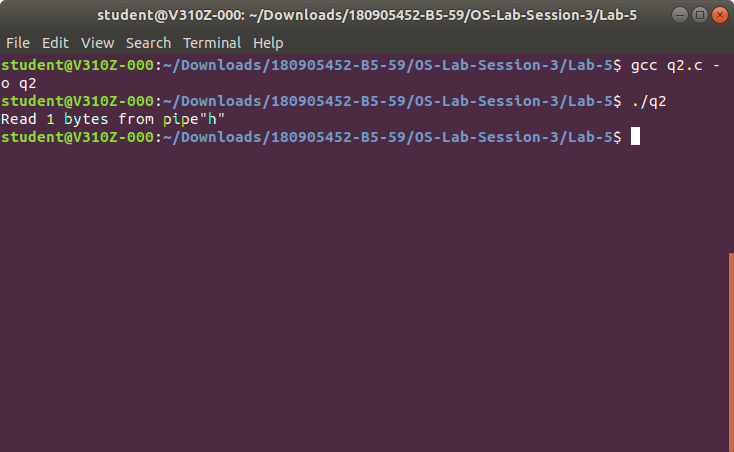
else

perror("Read");

exit(0);

}

**Output**



1. Write a C program to implement one side of FIFO.

**User 1**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<limits.h>

#include<fcntl.h>

#include<sys/msg.h>

#include<sys/stat.h>

#include<string.h>

#define FIFO\_NAME "my\_fifo"

#define BUFFER\_SIZE 10000

int main(int argc, char \*argv[])

{

int pipe\_fd;

int res;

int open\_mode1=O\_WRONLY;

int open\_mode2=O\_RDONLY;

int n=0;

char buffer[BUFFER\_SIZE+1];

if(access(FIFO\_NAME,F\_OK)==-1)

{

res=mkfifo(FIFO\_NAME,0777);

if(res!=0)

{

fprintf(stderr, "Could not create file%s\n",FIFO\_NAME);

exit(EXIT\_FAILURE);

}

}

printf("You can start chatting with User 2 now\n");

while(1)

{

pipe\_fd=open(FIFO\_NAME,open\_mode2);

printf("\nText from User 1: ");

res=read(pipe\_fd,buffer,BUFFER\_SIZE);

printf("%s\n",buffer );

close(pipe\_fd);

printf("Wait for User 1 reply\n");

pipe\_fd=open(FIFO\_NAME,open\_mode1);

printf("\nEnter Text to send User 1: ");

fgets(buffer,BUFFER\_SIZE,stdin);

res=write(pipe\_fd,buffer,BUFFER\_SIZE);

close(pipe\_fd);

}

(void)close(pipe\_fd);

printf("Process %d Finished\n",getpid());

exit(EXIT\_SUCCESS);

}

**User 2**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<limits.h>

#include<fcntl.h>

#include<sys/msg.h>

#include<sys/stat.h>

#include<string.h>

#define FIFO\_NAME "my\_fifo"

#define BUFFER\_SIZE 10000

int main(int argc, char \*argv[])

{

int pipe\_fd;

int res;

int open\_mode1=O\_WRONLY;

int open\_mode2=O\_RDONLY;

int n=0;

char buffer[BUFFER\_SIZE+1];

if(access(FIFO\_NAME,F\_OK)==-1)

{

res=mkfifo(FIFO\_NAME,0777);

if(res!=0)

{

fprintf(stderr, "Could not create file%s\n",FIFO\_NAME );

exit(EXIT\_FAILURE);

}

}

printf("You can start chatting with User 2 now\n");

while(1)

{

pipe\_fd=open(FIFO\_NAME,open\_mode1);

printf("\nEnter Text to send User 2: ");

fgets(buffer,BUFFER\_SIZE,stdin);

res=write(pipe\_fd,buffer,BUFFER\_SIZE);

close(pipe\_fd);

printf("Wait for User 2 reply\n");

pipe\_fd=open(FIFO\_NAME,open\_mode2);

printf("\nText from User 2: ");

res=read(pipe\_fd,buffer,BUFFER\_SIZE);

printf("%s\n",buffer );

close(pipe\_fd);

}

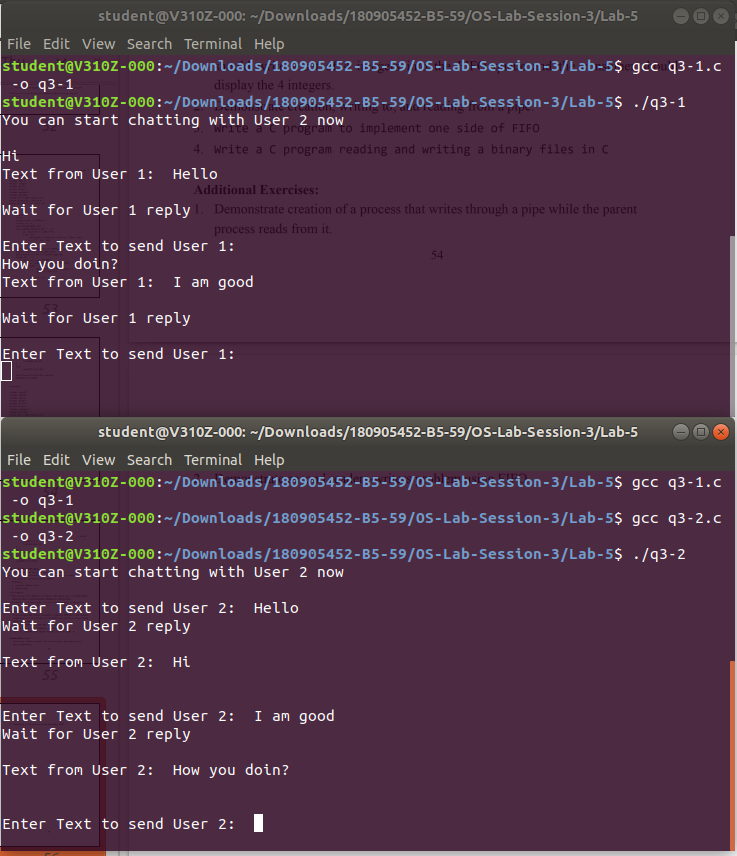
(void)close(pipe\_fd);

printf("Process %d Finished\n",getpid() );

exit(EXIT\_SUCCESS);

}

**Output**



4.) Write a C program reading and writing a binary file in C.

#include<stdio.h>

#include<stdlib.h>

int main()

{

FILE\* fptr;

int num=0;

fptr=fopen("demo.bin","wb+");

printf("Enter some numbers : \n");

for(int i=0;i<4;i++)

{

scanf("%d",&num);

fwrite(&num,sizeof(int),1,fptr);

}

printf("Writing done!\n");

fclose(fptr);

fptr=fopen("demo.bin","rb");

for(int i=0;i<4;i++)

{

fread(&num,sizeof(int),1,fptr);

printf("%d\n",num);

}

}

**Output**

