OS LAB 5

Aniruddha Amit Dutta

Roll - 58

180905488

Q1.

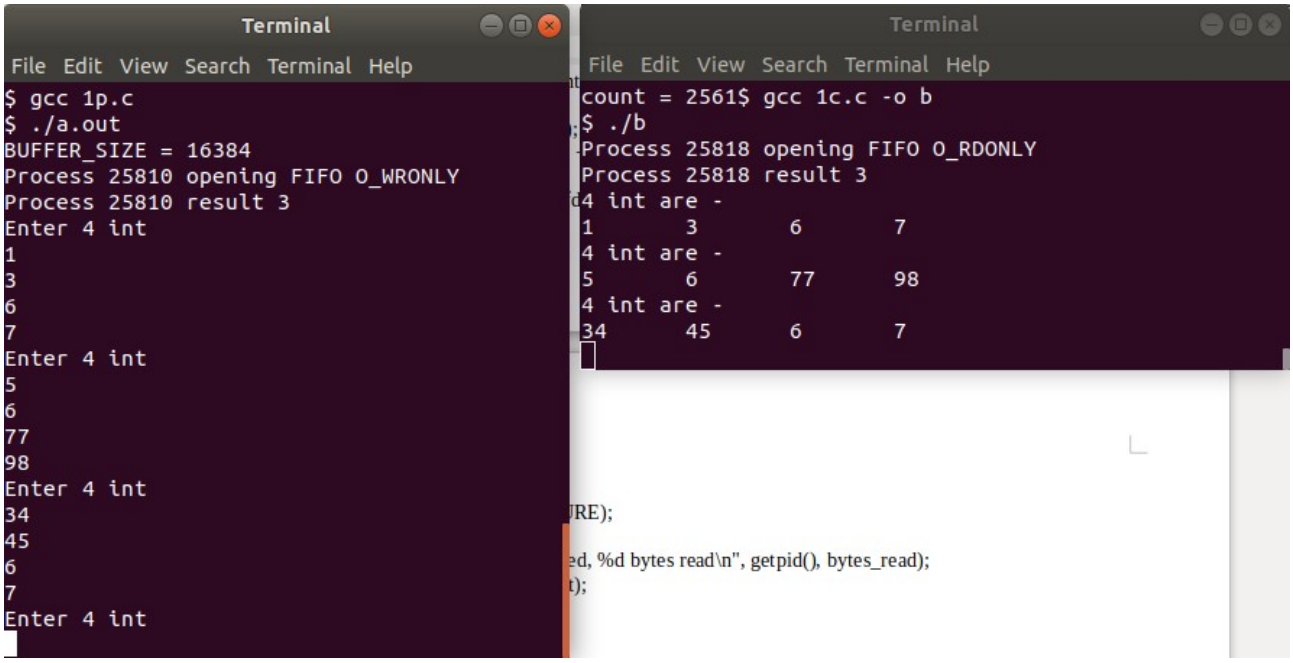
// producer

```
#include<unistd.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include<fcntl.h>
#include<limits.h>
#include<sys/types.h>
#include<sys/stat.h>
#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE (PIPE_BUF * 4)
#define TEN_MEG (1024 * 1024 * 10)
int main()
{
	int pipe_fd;
	int res;
	int open_mode = O_WRONLY;
	int bytes_sent = 0;
	printf("BUFFER_SIZE = %d\n",BUFFER_SIZE );
	// PIPE_BUF = 4096 -> 4 bytes = sizeof(int)
	int buffer[BUFFER_SIZE + 1];
	if (access(FIFO_NAME, F_OK) == -1) {
		res = mkfifo(FIFO_NAME, 0777);
		if (res != 0) {
			fprintf(stderr, "Could not create fifo %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) {
		while(bytes_sent < TEN_MEG) {
			printf("Enter 4 int \n");
			for (int i = 0; i < 4; ++i)
```

```
                {
                        scanf("%d",&buffer[i]);
                }
                res = write(pipe_fd, buffer, BUFFER_SIZE);
                if (res == -1) {
                        fprintf(stderr, "Write error on pipe\n");
                        exit(EXIT_FAILURE);
                }
                bytes_sent += res;
        }
        (void)close(pipe_fd);
    }
    else {
        exit(EXIT_FAILURE);
    }
    printf("Process %d finished\n", getpid());
    exit(EXIT_SUCCESS);
}

// consumer

#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <fcntl.h>
#include <limits.h>
#include <sys/types.h>
#include <sys/stat.h>
#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE (PIPE_BUF*4)
int count=0;
int main()
{
    int pipe_fd;
    int res;
    int open_mode = O_RDONLY;
    int buffer[BUFFER_SIZE + 1];
    int bytes_read = 0;
    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); ++count;
                printf("4 int are -\n");
                for (int i = 0; i < 4; ++i)
                {
                        printf("%d\t", buffer[i]);
                }
                printf("\n");
```

```
                bytes_read += res;
            } while (res > 0);
            (void)close(pipe_fd);
        }
        else {
            exit(EXIT_FAILURE);
        }
        printf("Process %d finished, %d bytes read\n", getpid(), bytes_read);
        printf("count = %d",count);
        exit(EXIT_SUCCESS);
}
```

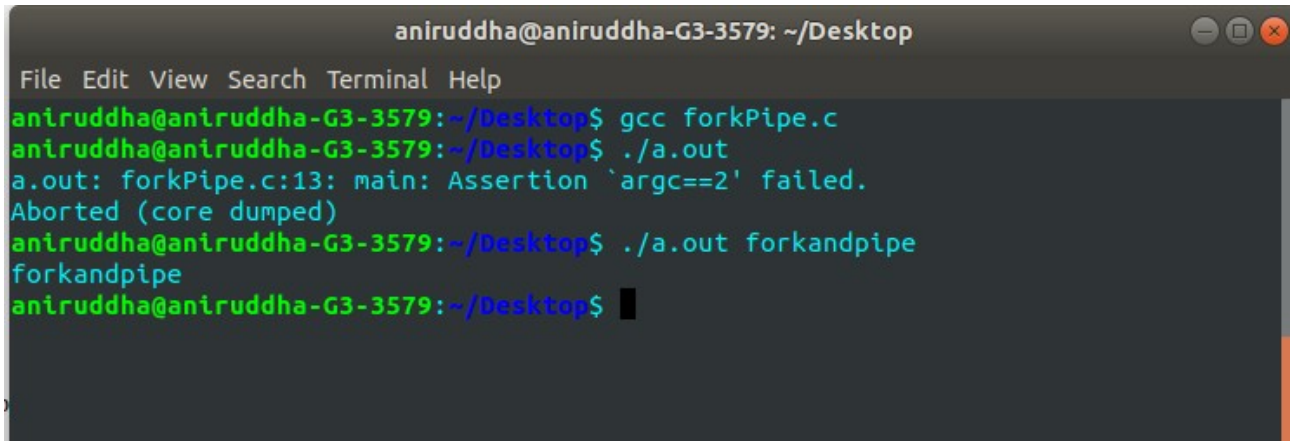
output -



Q2.

```
#include<unistd.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include<assert.h>
#include<sys/wait.h>

int main(int argc, char const *argv[])
{
        int pfd[2];
        pid_t cpid;
        char buf;
        assert(argc==2);
            if(pipe(pfd)==-1)
            {
                perror("pipe");
```

```
                        exit(EXIT_FAILURE);
                }
        cpid = fork();
        if(cpid==-1)
        {
                perror("fork");
                exit(EXIT_FAILURE);
        }


        if (cpid==0)
        {
                close(pfd[1]);
                while(read(pfd[0],&buf,1)>0)
                        write(STDOUT_FILENO,&buf,1);
                write(STDOUT_FILENO,"\n",1);
                close(pfd[0]);
                exit(EXIT_SUCCESS);
        }
        else
        {
                close(pfd[0]);
                write(pfd[1],argv[1],strlen(argv[1]));
                close(pfd[1]);
                wait(NULL);
                exit(EXIT_SUCCESS);
        }
}
```



Q3.

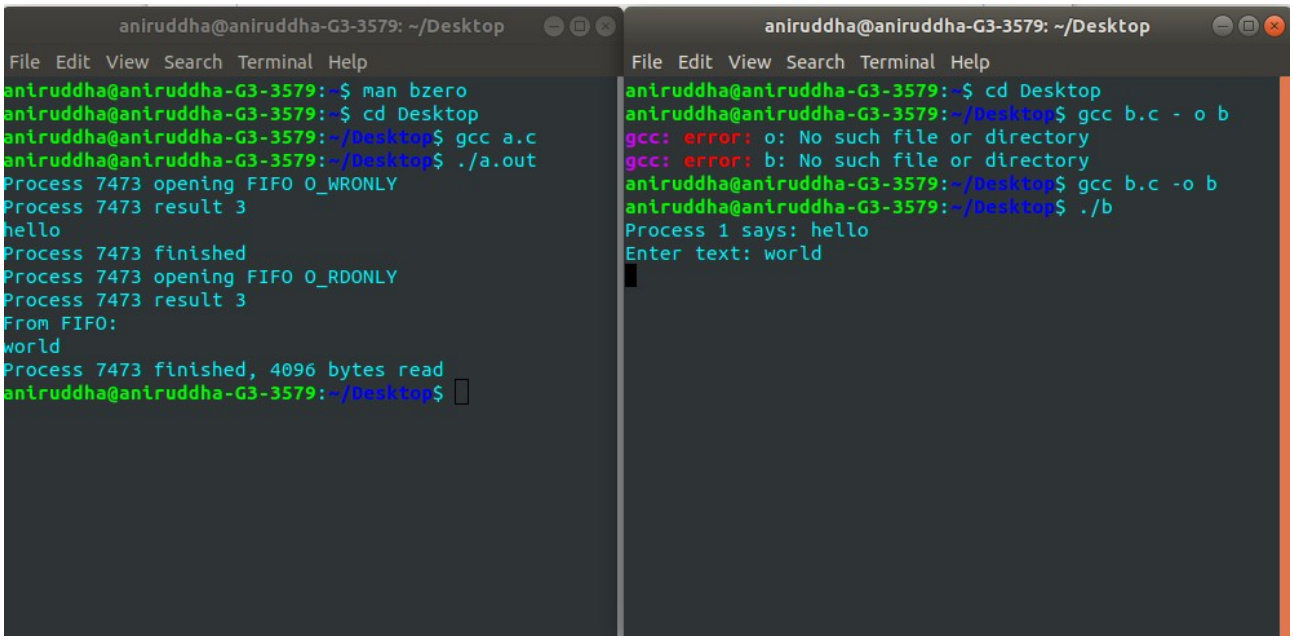
```
// write

#include<unistd.h>
#include<stdlib.h>
#include<stdio.h>
```

```
#include<string.h>
#include<fcntl.h>
#include<limits.h>
#include<sys/types.h>
#include<sys/stat.h>
#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE PIPE_BUF
#define TEN_MEG (4096)

int main()
{
	int pipe_fd;
	int res;
	int open_mode=O_WRONLY;
	int bytes_sent=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 fifo %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);
	res=0;
	if(pipe_fd!=-1)
	{

		scanf("%s",buffer);
		res=write(pipe_fd,buffer,BUFFER_SIZE);
		if(res==-1)
		{
			fprintf(stderr, "Write error on pipe\n");
			exit(EXIT_FAILURE);
		}
		bytes_sent+=res;

		(void)close(pipe_fd);
	}
	else
	{
		exit(EXIT_FAILURE);
	}

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


	open_mode=O_RDONLY;
```

```
	int bytes_read=0;
	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);
			bytes_read+=res;
		}while(res>0);
		(void)close(pipe_fd);
	}
	else
	{
		exit(EXIT_FAILURE);
	}
	printf("From FIFO:\n%s\n", buffer);
	printf("Process %d finished, %d bytes read\n", getpid(), bytes_read);


	exit(EXIT_SUCCESS);
}


// read

#include<unistd.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include<fcntl.h>
#include<limits.h>
#include<sys/types.h>
#include<sys/stat.h>
#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE PIPE_BUF
#define TEN_MEG (1024*1024*10)

int main()
{
	while(1){
	int pipe_fd;
	int res=0;
	int open_mode=O_RDONLY;
	int bytes_read=0;
	char buffer1[BUFFER_SIZE+1];
	memset(buffer1,'\0',sizeof(buffer1));

	pipe_fd=open(FIFO_NAME,open_mode);
```

```
        if(pipe_fd!=-1)
        {
                do
                {
                        res=read(pipe_fd,buffer1,BUFFER_SIZE);
                        bytes_read+=res;
                }while(res==0);
                (void)close(pipe_fd);
                printf("Process 1 says: %s\n", buffer1);
                if(strcmp(buffer1,"exit")==0)
                        break;
        }
        else
        {
                exit(EXIT_FAILURE);
        }

        res=0;
        open_mode=O_WRONLY;
        int bytes_sent=0;
        char buffer2[BUFFER_SIZE+1];

        pipe_fd=open(FIFO_NAME,open_mode);

        if(pipe_fd!=-1)
        {

                printf("Enter text: ");
                scanf("%s",buffer2);
                res=write(pipe_fd,buffer2,BUFFER_SIZE);
                if(res==-1)
                {
                        fprintf(stderr, "Write error on pipe\n");
                        exit(EXIT_FAILURE);
                }
                bytes_sent+=res;


                (void)close(pipe_fd);
                if(strcmp(buffer2,"exit")==0)
                        break;
        }
        else
        {
                exit(EXIT_FAILURE);
        }
        }
}
```

output ->



Q4.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>


int main(int argc, char **argv){
	int fd1;
	int n;
	fd1 = open("example.bin",O_RDWR|O_CREAT,S_IRWXU|S_IRUSR|S_IWUSR);

	char buf[10];
	printf("enter string \n");
	scanf("%s",buf);

	printf("buf = %s",buf);

	printf("write to file \n");
	if( write(fd1,buf,sizeof(buf)) < 0 ){
		perror("write");
	}
	if( ( n=read(fd1,buf,10) ) >=0 ){
		buf[n]='\0'; /* terminate the string*/
		printf("read %d bytes from the file : ",n);
```

```
        }
        else{
                perror("read");
        }
        exit(0);
}
```

```
$ gcc 4.c
$ ./a.out
enter string
HELLOWORLD
buf = HELLOWORLDwrite to file
read 0 bytes from the file : $ cat example.bin
HELLOWORLD$
```