VVIT
VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY
Permanently Affiliated to JNTU Kakinada, Approved by AICTE
Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified
Nambur, Pedakakani (M), Guntur (Dt) - 522508
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
B.Tech Program is Accredited by NBA

CERTIFICATE

Name of the Lab	: OPERATING SYSTEMS
Name of the Student	: Thota Nagababu
Student Regd. No.	: 18BQ1A05K3
CLASS	: III B.TECH. I SEM CSE – D

GithubLink:https://github.com/nagababuthota984/5K3-OS-LAB

VVIT
VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY
Permanently Affiliated to JNTU Kakinada, Approved by AICTE
Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified
Nambur, Pedakakani (M), Guntur (Dt) - 522508
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
B.Tech Program is Accredited by NBA

INDEX

S. No.	Name of the Experiment	LAST DATE	PAGE NO
5B	CREATING A ONE WAY PIPE USING SINGLE PROCESS	JAN 12	3-5
	ii)CREATING CHILD & PARENT PROCESS AND MAKING THEM COMMUNICATE BY USING PIPE()	JAN 12	5-7
	iii)CREATING A TWO-WAY PIPES BETWEEN TWO	JAN 12	7-9

Experiment 5B (i)

AIM: Create a one way pipe using a single process

DESCRIPTION:

Pipe is a communication medium between two or more related or interrelated processes. It can be either within one process or a communication between the child and the parent processes. Communication can also be multi-level such as communication between the parent, the child and the grand-child, etc. Communication is achieved by one process writing into the pipe and other reading from the pipe. To achieve the pipe system call, create two files, one to write into the file and another to read from the file.

Pipe mechanism can be viewed with a real-time scenario such as filling water with the pipe into some container, say a bucket, and someone retrieving it, say with a mug. The filling process is nothing but writing into the pipe and the reading process is nothing but retrieving from the pipe. This implies that one output (water) is input for the other (bucket).

LIBRARIES:

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

#include <unistd.h>

CODE:

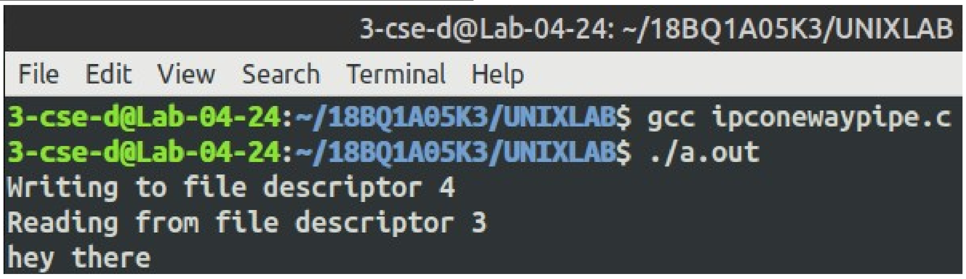
1.IPCONEWAYSERVER

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
int main(void)
{
	int pdfs[2];
	char buf[30];
	if(pipe(pdfs)==-1)
	{
		perror("pipe");
		exit(1);
	}
	printf("Writing to file descriptor %d\n",pdfs[1]);
	write(pdfs[1],"hey there",9);
	printf("Reading from file descriptor %d\n",pdfs[0]);
	read(pdfs[0],buf,9);
	printf("%s\n",buf);
	return 0;
}
```

OUTPUT:

```
writing to the file descriptor #4
reading from file descriptor #3
hey there
```

OUTPUT SCREENSHOTS:


3-cse-d@Lab-04-24: ~/18BQ1A05K3/UNIXLAB
File Edit View Search Terminal Help

```
3-cse-d@Lab-04-24:~/18BQ1A05K3/UNIXLAB$ gcc ipconewaypipe.c
3-cse-d@Lab-04-24:~/18BQ1A05K3/UNIXLAB$ ./a.out
Writing to file descriptor 4
Reading from file descriptor 3
hey there
```

AIM:ii) Creating Child & Parent Process and making them communicate using pipe()

DESCRIPTION:

Pipe is a communication medium between two or more related or interrelated processes. It can be either within one process or a communication between the child and the parent processes. ... To achieve the pipe system call, create two files, one to write into the file and another to read from the file.

LIBRARIES USED :

#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/wait.h>
#include <unistd.h>

CODE:

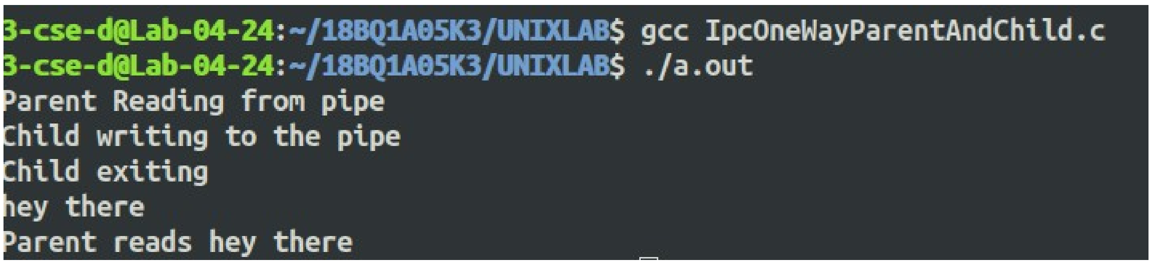
2.IPCONEWAYPIPE PARENT AND CHILD

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/wait.h>
#include <unistd.h>
int main(void)
{
        int pdfs[2];
        char buf[30];
        pipe(pdfs);
        if(!fork())
        {
                printf("Child writing to the pipe\n");
                write(pdfs[1],"hey there",9);
                printf("Child exiting\n");
                exit(1);
        }
        else
        {
                printf("Parent Reading from pipe \n");
                read(pdfs[0],buf,9);
                printf("%s\n",buf);
                printf("Parent reads %s \n",buf);
                wait(NULL);
        }
        return 0;
}
```

OUTPUT:

```
Parent reading from pipe
child writing to the pipe
child exiting
hey there
parent reads hey there
```

OUTPUT SCREENSHOTS:



```
3-cse-d@Lab-04-24:~/18BQ1A05K3/UNIXLAB$ gcc IpcOneWayParentAndChild.c
3-cse-d@Lab-04-24:~/18BQ1A05K3/UNIXLAB$ ./a.out
Parent Reading from pipe
Child writing to the pipe
Child exiting
hey there
Parent reads hey there
```

AIM:

iii) CREATING A TWO-WAY PIPES BETWEEN TWO PROCESS

DESCRIPTION:

Pipe is a communication medium between two or more related or interrelated processes. It can be either within one process or a communication between the child and the parent processes.

There will be 2 pipes and 2 processes parent and child and parent process will write onto pipe 1 and child process will be reading from it.

And on another pipe child process will be writing and parent process will be reading from it.

LIBRARIES USED:

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

CODE:

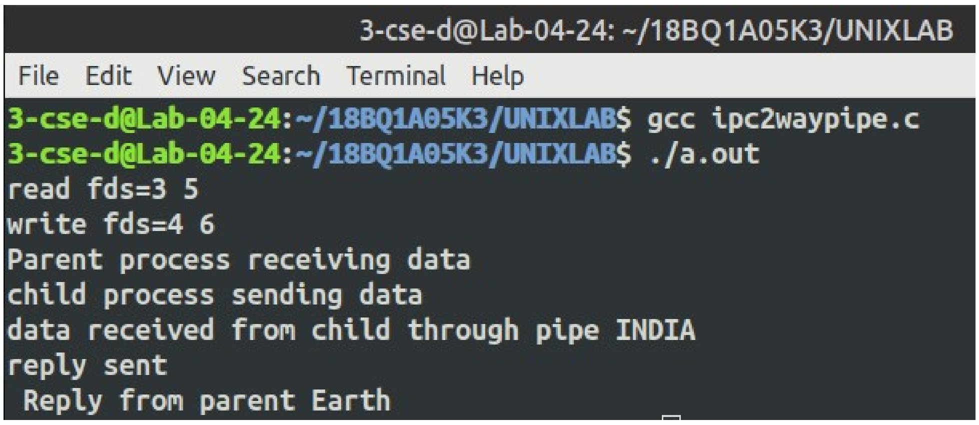
3.ipc2way pipes

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
int main()
{
        int p1[2],p2[2],n,pid;
        char buf1[25],buf2[25];
        pipe(p1),pipe(p2);
        printf("read fds=%d %d \n",p1[0],p2[0]);
        printf("write fds=%d %d \n",p1[1],p2[1]);
        pid=fork();
        if(pid==0)
        {
                close(p1[0]);
                printf("child process sending data \n ");
                write(p1[1],"INDIA",6);
                close(p2[1]);
                read(p2[0],buf1,25);
                printf("Reply from parent %s \n ",buf1);
                sleep(2);
        }
        else
        {
                close(p1[1]);
                printf("Parent process receiving data\n");
                n=read(p1[0],buf2,sizeof(buf2));
                printf("data received from child through pipe
%s \n",buf2);
                sleep(3);
                close(p2[0]);
                write(p2[1],"Earth",6);
                printf("reply sent\n");
        }
}
```

OUTPUT:

```
readfds =3 5
writefds=4 6
Parent process receiving data
Childprocess sending data
Data received from child through pipe INDIA
reply sent
Reply from parent Earth
```

OUTPUT SCREENSHOTS:


3-cse-d@Lab-04-24: ~/18BQ1A05K3/UNIXLAB
File Edit View Search Terminal Help

```
3-cse-d@Lab-04-24:~/18BQ1A05K3/UNIXLAB$ gcc ipc2waypipe.c
3-cse-d@Lab-04-24:~/18BQ1A05K3/UNIXLAB$ ./a.out
read fds=3 5
write fds=4 6
Parent process receiving data
child process sending data
data received from child through pipe INDIA
reply sent
 Reply from parent Earth
```