Ex03-Linux IPC - Pipes

AIM:

To write a C program that illustrate communication between two process using unnamed and named pipes

DESIGN STEPS:

Step 1:

Navigate to any Linux environment installed on the system or installed inside a virtual environment like virtual box/vmware or online linux JSLinux (https://bellard.org/jslinux/vm.html?url=alpine-x86.cfg&mem=192) or docker.

Step 2:

Write the C Program using Linux Process API - pipe(), fifo()

Step 3:

Testing the C Program for the desired output.

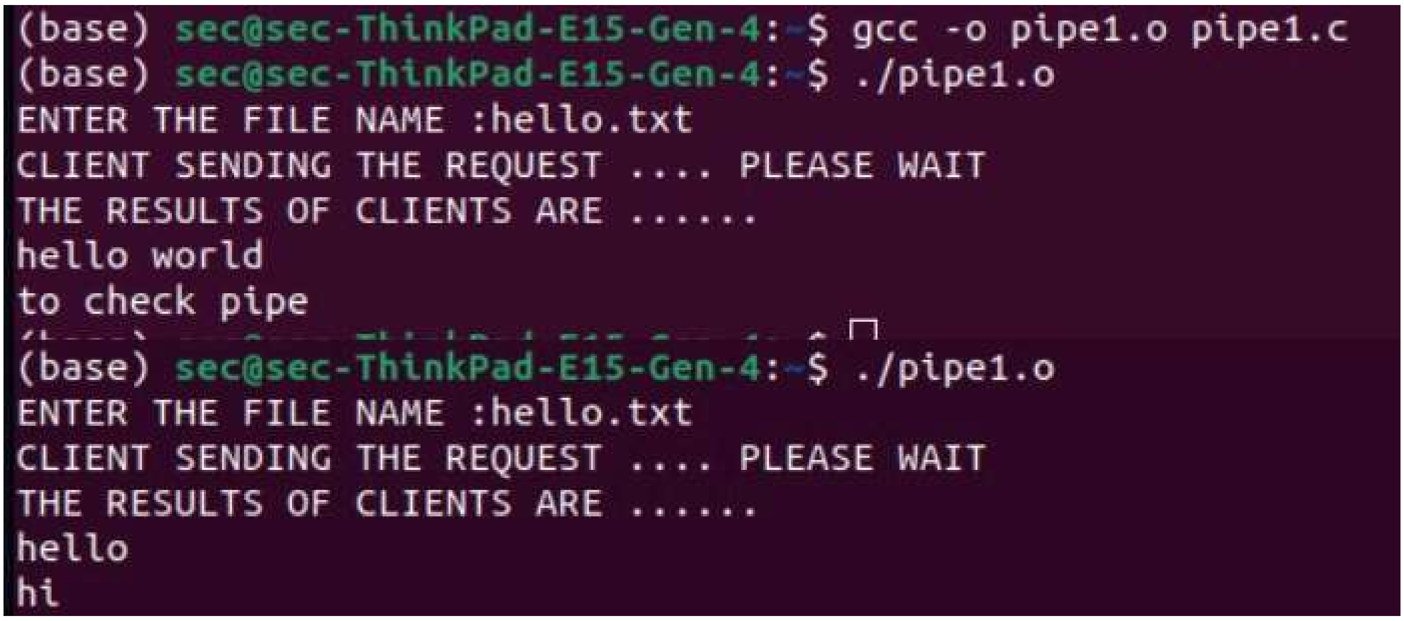
PROGRAM:

C Program that illustrate communication between two process using unnamed pipes using Linux API system calls

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<string.h>
#include<fcntl.h>
#include<unistd.h>
#include<sys/wait.h>
void server(int,int);
void client(int,int);
int main()
{
int p1[2],p2[2],pid, *waits;
pipe(p1);
pipe(p2);
pid=fork();
if(pid==0) {
close(p1[1]);
close(p2[0]);
server(p1[0],p2[1]); return 0;
 }
close(p1[0]);
close(p2[1]);
client(p1[1],p2[0]);
wait(waits);
return 0;
}
```

```
void server(int rfd,int wfd)
{
int i,j,n;
char fname[2000];
char buff[2000];
n=read(rfd,fname,2000);
fname[n]='\0';
int fd=open(fname,O_RDONLY);
sleep(10);
if(fd<0)
write(wfd,"can't open",9);
else
n=read(fd,buff,2000);
write(wfd,buff,n);
}
void client(int wfd,int rfd) {
int i,j,n; char fname[2000];
char buff[2000];
printf("ENTER THE FILE NAME :");
scanf("%s",fname);
printf("CLIENT SENDING THE REQUEST .... PLEASE WAIT\n");
sleep(10);
write(wfd,fname,2000);
n=read(rfd,buff,2000);
buff[n]='\0';
printf("THE RESULTS OF CLIENTS ARE ...... \n"); write(1,buff,n);
}
```

OUTPUT



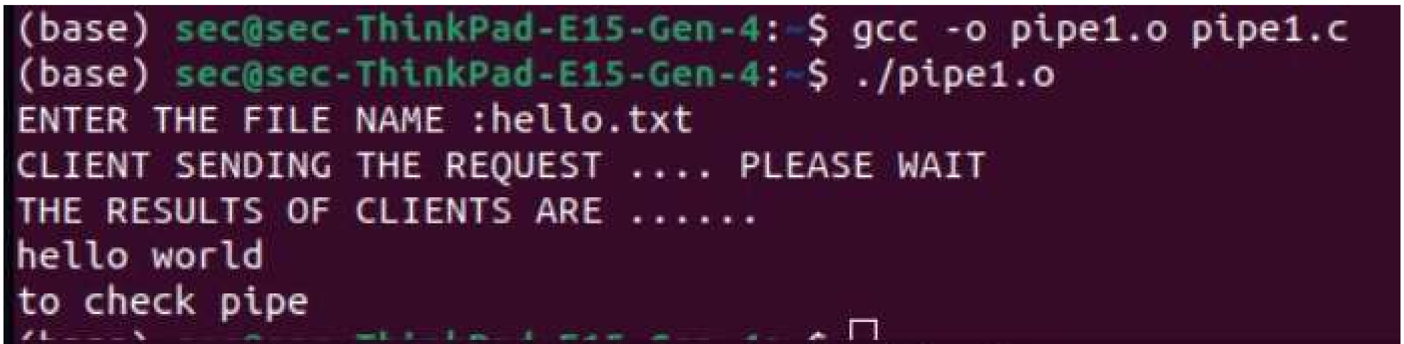
```
(base) sec@sec-ThinkPad-E15-Gen-4:~$ gcc -o pipe1.o pipe1.c
(base) sec@sec-ThinkPad-E15-Gen-4:~$ ./pipe1.o
ENTER THE FILE NAME :hello.txt
CLIENT SENDING THE REQUEST .... PLEASE WAIT
THE RESULTS OF CLIENTS ARE ......
hello world
to check pipe
(base) sec@sec-ThinkPad-E15-Gen-4:~$ ./pipe1.o
ENTER THE FILE NAME :hello.txt
CLIENT SENDING THE REQUEST .... PLEASE WAIT
THE RESULTS OF CLIENTS ARE ......
hello
hi
```

C Program that illustrate communication between two process using named pipes using Linux API system calls

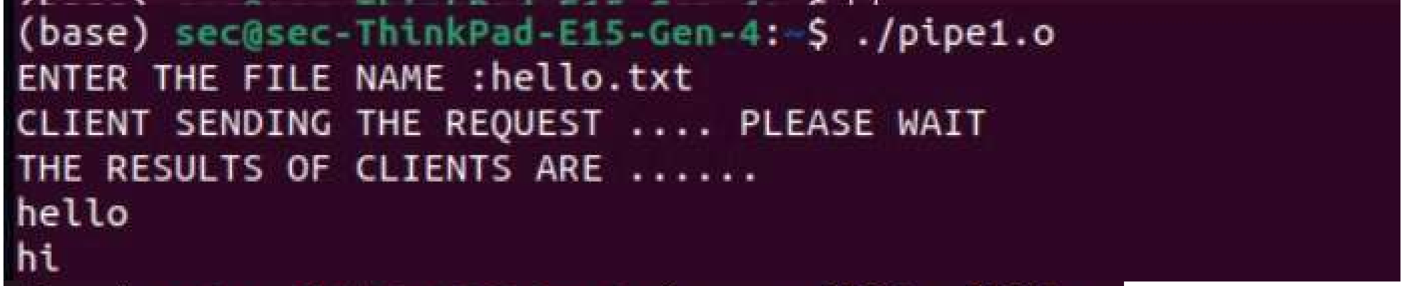
```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
```

```
#include <sys/stat.h>
int main(){
int res = mkfifo("/tmp/my_fifo", 0777);
if (res == 0) printf("FIFO created\n");
exit(EXIT_SUCCESS);
}
```

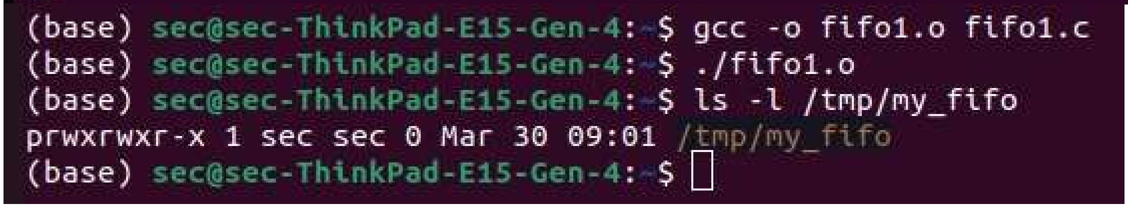
OUTPUT



```
(base) sec@sec-ThinkPad-E15-Gen-4:~$ gcc -o pipe1.o pipe1.c
(base) sec@sec-ThinkPad-E15-Gen-4:~$ ./pipe1.o
ENTER THE FILE NAME :hello.txt
CLIENT SENDING THE REQUEST .... PLEASE WAIT
THE RESULTS OF CLIENTS ARE ......
hello world
to check pipe
```



```
(base) sec@sec-ThinkPad-E15-Gen-4:~$ ./pipe1.o
ENTER THE FILE NAME :hello.txt
CLIENT SENDING THE REQUEST .... PLEASE WAIT
THE RESULTS OF CLIENTS ARE ......
hello
hi
```



```
(base) sec@sec-ThinkPad-E15-Gen-4:~$ gcc -o fifo1.o fifo1.c
(base) sec@sec-ThinkPad-E15-Gen-4:~$ ./fifo1.o
(base) sec@sec-ThinkPad-E15-Gen-4:~$ ls -l /tmp/my_fifo
prwxrwxr-x 1 sec sec 0 Mar 30 09:01 /tmp/my_fifo
(base) sec@sec-ThinkPad-E15-Gen-4:~$ 
```

RESULT:

The program is executed successfully.