Name: Aditya Somani

Roll No: T1851061

<u>Div</u>: A

## **ASSIGNMENT NO. 7b**

**TITLE:** Inter-process Communication (IPC) in Linux using FIFO.

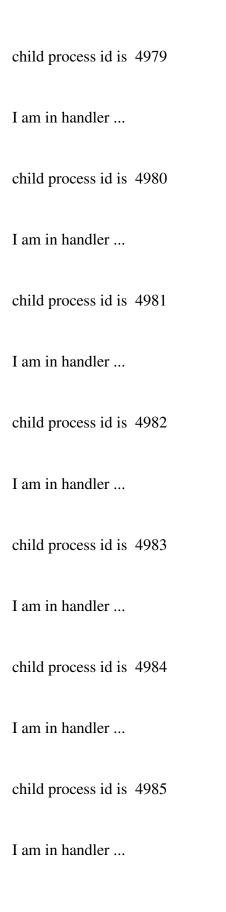
```
#include<stdio.h>
#include<fcntl.h>
#include<sys/types.h>
#include<unistd.h>
#include<sys/stat.h>
int main()
{
        int fd,i=0,fd1;
        FILE * fp;
        char * myfifo = "/tmp/myfifo";
        char * myfifo1 = "/tmp/myfifo1";
        char buf[100],temp,buffer[100];
        mkfifo(myfifo,0666);
        mkfifo(myfifo1,0666);
       fd=open(myfifo,O_WRONLY);
        printf("Enter your message \n type z to terminate \n");
       while(1)
       {
               scanf("%c",&temp);
```

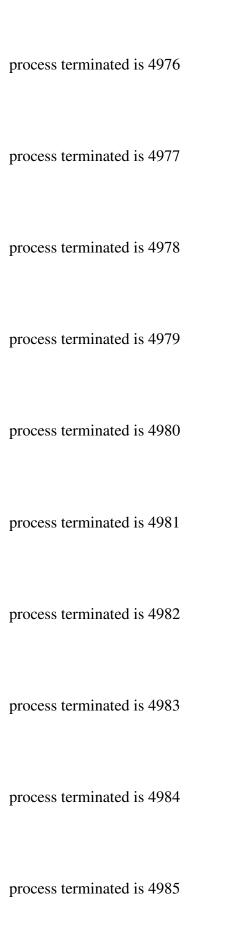
```
if(temp != 'z')
                   buf[i]=temp;
                   i++;
            }
            else
            break;
      }
      write(fd,buf,sizeof(buf));
      sleep(5);
      close(fd);
      fd1=open(myfifo1,O_RDONLY);
      read(fd1,buffer,100);
      printf(" %s",buffer);
      //close(fd1);
      unlink(myfifo);
return 0;
}
rohan@rohan:~$ cd Desktop
rohan@rohan:~/Desktop$ gccfifo.c
rohan@rohan:~/Desktop$ ./a.out
Enter your message
type z to terminate
ROHAN IOIT
```

```
TE IT
Ζ
Z
Number of character: 18.
Number of words: 6.
Number of lines: 3
rohan@rohan:~$ cd Desktop
rohan@rohan:~/Desktop$ gcc fifo1.c
rohan@rohan:~/Desktop$ ./a.out
       /*====Inter process communication using Signals====*/
# include <unistd.h>
# include <sys/types.h>
# include <stdio.h>
# include <sys/wait.h>
# include <signal.h>
# include <stdlib.h>
void handler(int sig)
{
       printf("\n\tI am in handler ...\n");
}
int main()
{
```

```
int status,i;
pid_tpid[10];
struct sigaction act;
//act.sa_flags=SA_NOCLDSTOP;
act.sa_handler=handler;
sigaction(SIGCHLD,&act,NULL);
for(i=0;i<10;i++)
{
       pid[i]=fork();
       if(pid[i] < 0)
               printf("Error while creating child process...!!!");
               exit(0);
        }
       else if(pid[i]==0)
        {
               printf("\n\tchild process id is %d\n",getpid());
               return 1;
        }
       else
               sleep(2);
        }
}
for(i=0;i<10;i++)
```

```
{
            pid[i] = wait(&status);
            printf("\n\tprocess terminated is %d\n\n",pid[i]);
      }
     printf("Parent Process with pid=%d ...\n",getppid());
}
rohan@rohan:~$ cd Desktop
rohan@rohan:~/Desktop$ gccsignals.c
rohan@rohan:~/Desktop$ ./a.out
      child process id is 4976
      I am in handler ...
     child process id is 4977
      I am in handler ...
      child process id is 4978
      I am in handler ...
```





Parent Process with pid=4955 ...

rohan@rohan:~/Desktop\$