2022PMN4226
Mayank Lakhotia

Practical 2

Using the "pipe ()" system call, implement the following: -

system calls used:

a) write (file_descriptor, our meessage, length of out message)

b) read (file_descriptor , our message, length of our output)

//value of file_descriptor =0 (standard input device , 1 sytandard output device , 2 is for error)

c)pipe()- creates a unidirectional pipe

- writing end arr[1]
- reading end arr[0]

d) fork() - returns

- 0 child ID
- 1 parent ID
- -1 when there is an error

(1) Perform inter-process communication between a Parent and Child process.

Source Code: -

```
#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<unistd.h>
```

```
#include<sys/wait.h>
#include<errno.h>
int main(int argc, char* argv[]){
int fd[2];
if(pipe(fd) == -1){
printf("An error occured with opening the pipe\n");
return 1;
}
int id = fork();
if(id==0){
close(fd[0]);
int x;
printf("Input a no.");
scanf("%d",&x);
if (write(fd[1], &x, sizeof(int)) == -1){
printf("An error occured with  the pipe\n");
return 2;
}
close(fd[1]);
}else{
close(fd[1]);
int y;
```

```
read(fd[0], &y, sizeof(int));

close(fd[0]);

printf("Got from child Process %d\n",y);

}

return 0;

}
```

Output: -

```
mayank@mayank-virtual-machine:~/Desktop/parent child communication$ ./test
Input a no.10
Got from child Process 10
mayank@mayank-virtual-machine:~/Desktop/parent child communication$
```

(2) Perform inter-process communication between TWO Child processes.

Source Code: -

```
#include <stdio.h>

#include <unistd.h>


int main()

{

int pipefds[2];

int isPoss;

int f1, f2;
```

```
char writemsg1[20];
char writemsg2[20];
printf("enter first message\n");
fgets(writemsg1,sizeof(writemsg1),stdin);
printf("enter second message\n");
fgets(writemsg2,sizeof(writemsg2),stdin);
char Readmsg[20];
isPoss = pipe(pipefds);
if (isPoss == -1)
{
printf("Can not create pipe\n");
return 1;
}
f1 = fork();
f2 = fork();


if (f1 > 0)
{
read(pipefds[0], Readmsg, sizeof(Readmsg));
printf("Child1 Process - Reading from M1 is %s\n", Readmsg);
read(pipefds[0], Readmsg, sizeof(Readmsg));
```

```
printf("Child1 Process - Reading from M2 is %s\n", Readmsg);
}
else if (f2 > 0)
{
printf("Child2 Process - Writing to pipe M1 is %s\n", writemsg1);
write(pipefds[1], writemsg1, sizeof(writemsg1));
printf("Child2 Process - Writing to pipe M2 is %s\n", writemsg2);
write(pipefds[1], writemsg2, sizeof(writemsg2));
}
return 0;
}
```

Output: -

```
mayank@mayank-virtual-machine:~/Desktop/child to child$ ./test
enter first message
Hi
enter second message
Hello
Child2 Process - Writing to pipe M1 is Hi

Child2 Process - Writing to pipe M2 is Hello

Child1 Process - Reading from M1 is Hi

Child1 Process - Reading from M2 is Hello
```