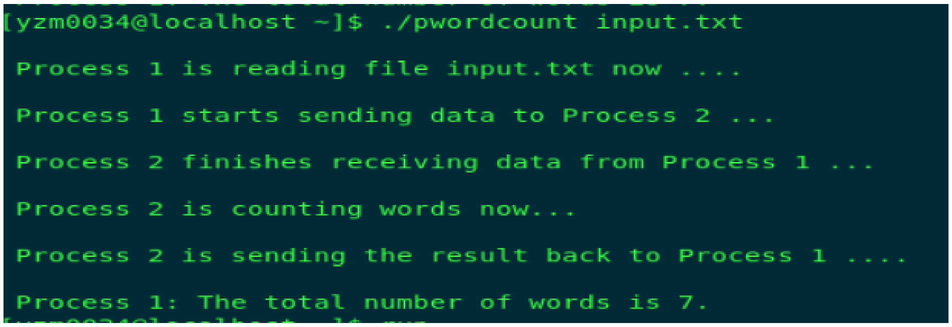
Report

The object of Project 2 is to create a communication channel between two processes using UNIX pipes. Pipes are the IPC (Inter Process Communication) communication mechanisms which is efficient way collaborating between multiple processes. In this project, two pipes were created between two processor for communication in producer-customer fashion.

The first process which is also called a Parent process, which read a text file as an input. After Parent process accepting input file, the file is then send to second process i.e. child process. The function of child process to count the number of words in the document. After counting the words in the document is then send back to parent process. The parent process publishing the result to the user. The communication between user and the wordcount tool occurs at command line. Below is the snapshot of the command line communication between user and wordcount tool.



```
[yzm0034@localhost ~]$ ./pwordcount input.txt

 Process 1 is reading file input.txt now ....

 Process 1 starts sending data to Process 2 ...

 Process 2 finishes receiving data from Process 1 ...

 Process 2 is counting words now...

 Process 2 is sending the result back to Process 1 ....

 Process 1: The total number of words is 7.
```

The IPC communication using pipes are the efficient way of communicating between processes. The design of this is project is very simple yet efficient. First, we describe file descriptor (fd) for two end of the pipe for read and write operations. The fd direct the processes in the communication channel. Thereafter, we create two pipe for two way communication and check their connection using file descriptor. After successful conformation of pipe connectivity, we create fork(). The function of fork is to create a child process for parallel execution. Using fork we assign tasks for parent and child process. A child process is nothing but a subset of parent process and having all the functionality of parent. By now we have design the architecture of the wordcount tool.

Now, lets dive in the communication process. Ones the parent process fetch file from command line, the parent check the correct orientation of the file. If the file is appropriate then parent write to the pipe and send the file via pipe. Note that while writing the parent has closed all other unused pipe ports such reading end because parent process is writing at the moment and avoid performing two task at time. On the other end of the pipe, the child process accept the file input using read function. Note that the child has also close write end at for the pipe 1 as it is only used for reading. After fetching the file from parent the child process perform counting operation on the file. Ones finish the counting task the child use second pipe to transfer the counted words. The child now has closed the reading end on pipe 2 as he is writing to the pipe2. On the other end the parent receive the count words using pipe2. Now parent display the counted words on the command line. This is how the communication between two processes held inside the wordcount tools.