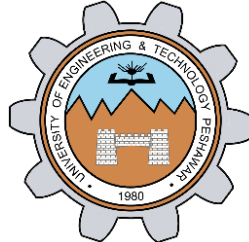


INTER-PROCESS COMMUNICATION

LAB # 10



Fall 2023

CSE-302L


Systems Programming Lab

Submitted by: **AIMAL KHAN**

Registration No.: **21PWCSE1996**

Class Section: **A**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: 

Submitted to:

Engr. Abdullah Hamid

Sunday, January 28, 2024

Department of Computer Systems Engineering
University of Engineering and Technology, Peshawar

CSE 302L: SYSTEMS PROGRAMMING LAB

LAB ASSESSMENT RUBRICS

Criteria & Point Assigned	Outstanding 2	Acceptable 1.5	Considerable 1	Below Expectations 0.5	Score
Attendance and Attentiveness in Lab PLO08	Attended in proper Time and attentive in Lab	Attended in proper Time but not attentive in Lab	Attended late but attentive in Lab	Attended late not attentive in Lab	
Capability of writing Program/Algorithm/Drawing Flow Chart PLO1, PLO2, PLO3, PLO5	Right attempt/ no errors and well formatted	Right attempt/ no errors but not well formatted	Right attempt/ minor errors and not well formatted	Wrong attempt	
Result or Output/ Completion of target in Lab PLO9	100% target has been completed and well formatted.	75% target has been completed and well formatted.	50% target has been completed but not well formatted.	None of the outputs are correct.	
Overall, Knowledge PLO10,	Demonstrates excellent knowledge of lab	Demonstrates good knowledge of lab	Has partial idea about the Lab and procedure followed	Has poor idea about the Lab and procedure followed	
Attention to Lab Report PLO4,	Submission of Lab Report in Proper Time i.e., in next day of lab, with proper documentation.	Submission of Lab Report in proper time but not with proper documentation.	Late Submission with proper documentation.	Late Submission very poor documentation	

Instructor:

Name: _____

Signature: _____

INTER-PROCESS COMMUNICATION

Objectives:

The objectives of this lab are to gain a practical understanding of key system programming concepts, including

- Pipes
- FIFOs

Tasks:

Task 1: A program in which a child writes a string to a pipe and the parent reads the string.

Code in C:

```
// A program in which a child writes a string to a pipe and the
parent reads the string.
```

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

#define MAX_BUFF_SIZE 128

int main()
{
    int fd[2];
    if (pipe(fd) == -1)
    {
        perror("Error While creating pipe.\n");
        return 1;
    }

    pid_t pid = fork();
    if (pid == -1)
    {
        perror("Error while creating childs.\n");
        return 1;
    }
    if(pid == 0){
        char *str = "Hello Baba\n";
        if(write(fd[1], str, strlen(str) + 1) == -1){
            perror("Error while writing to pipe.\n");
            return 1;
        }
    }
}
```

```

else{
    char buff[MAX_BUFF_SIZE];
    if(read(fd[0], buff, MAX_BUFF_SIZE) == -1){
        perror("Error while reading from pipe.\n");
        return 1;
    }
    printf("Parent received: %s", buff);
}

return 0;
}

```

Output:

```

Select hamza2002@DESKTOP-GRD25B9: /mnt/d/5th_SEMESTER/SP_LAB/lab10/tasks
hamza2002@DESKTOP-GRD25B9:/mnt/d/5th_SEMESTER/SP_LAB/lab10/tasks$ ./task1.o
Parent received: Hello Baba
hamza2002@DESKTOP-GRD25B9:/mnt/d/5th_SEMESTER/SP_LAB/lab10/tasks$ _

```

Task 2: Write a program that creates a process fan. Parent process writes to the pipe and all the child processes read the message from pipe and display it on stdout

Code in C:

```

// Write a program that creates a process fan. Parent process writes
to the pipe and all the child processes read the message from pipe
and display it on stdout.

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

#define MAX_BUFF_SIZE 128
#define CHILD_COUNT 3

int main()
{
    int fd[2], i;
    pid_t pid;
    char *str = "Hello Baba\n", buff[MAX_BUFF_SIZE];
    if (pipe(fd) == -1)
    {
        perror("Error While creating pipe.\n");
        return 1;
    }
    for (i = 0; i < CHILD_COUNT; i++){
        if ((pid = fork()) == -1)

```

```

    {
        perror("Error while creating childs.\n");
        return 1;
    }

    if (pid == 0)
    {
        if (read(fd[0], buff, MAX_BUFF_SIZE) == -1)
        {
            perror("Error while reading from pipe.\n");
            return 1;
        }
        printf("Child [%ld] received: %s", (long)(getpid()),
buff);
        break;
    }
    else
    {
        wait(NULL);
        if (write(fd[1], str, strlen(str) + 1) == -1)
        {
            perror("Error while writing to pipe.\n");
            return 1;
        }
    }
}

return 0;
}

```

Output:

```

hamza2002@DESKTOP-GRD25B9: /mnt/d/5th_SEMESTER/SP_LAB/lab10/tasks
hamza2002@DESKTOP-GRD25B9:/mnt/d/5th_SEMESTER/SP_LAB/lab10/tasks$ ./task2.o
hi
hellp
hello
kese ho yar
^C
hamza2002@DESKTOP-GRD25B9:/mnt/d/5th_SEMESTER/SP_LAB/lab10/tasks$

```

Task 3: Chatting between two process using FIFO

Code in C:

```

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>

int main(int argc, char *argv[]) {
    if (argc != 2) {

```

```

        fprintf(stderr, "Usage: %s <FIFO_NAME>\n", argv[0]);
        return 1;
    }

    const char *fifoName = argv[1];
    char buf[1024];
    ssize_t numBytes;

    // Create a FIFO if it does not exist
    if (mkfifo(fifoName, 0666) == -1) {
        perror("mkfifo");
        return 1;
    }

    // Fork a process
    pid_t pid = fork();
    if (pid == -1) {
        perror("fork");
        return 1;
    }

    if (pid == 0) { // Child process for writing to FIFO
        int fifoFd = open(fifoName, O_WRONLY);
        if (fifoFd == -1) {
            perror("open");
            return 1;
        }

        while (fgets(buf, sizeof(buf), stdin)) {
            write(fifoFd, buf, sizeof(buf));
        }

        close(fifoFd);
    } else { // Parent process for reading from FIFO
        int fifoFd = open(fifoName, O_RDONLY);
        if (fifoFd == -1) {
            perror("open");
            return 1;
        }

        while ((numBytes = read(fifoFd, buf, sizeof(buf))) > 0) {
            write(STDOUT_FILENO, buf, numBytes);
        }

        close(fifoFd);
    }

    return 0;}

```

Reference:

To view my codes, please refer to my GitHub account:
[https://github.com/aimalexe/DCSE/tree/main/semester 5 \(fall-23\)/systems programming lab/lab reports](https://github.com/aimalexe/DCSE/tree/main/semester%205%20(fall-23)/systems_programming_lab/lab_reports) .

Conclusion:

In summary, this laboratory experience has provided a comprehensive exploration of various fundamental system programming concepts, including pipes and FIFOs. These newfound insights are valuable assets that will enhance my problem-solving abilities and future project contributions.

The End.