Operating Systems Lab

Lab# 12

Submitted By
QASIM ALI (20P-0070)
Submitted to : M Ahsan
(INSTRUCTOR CS)

DEPARTMENT OF COMPUTER SCIENCE

FAST NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES, PESHAWAR

Session 2020-2024

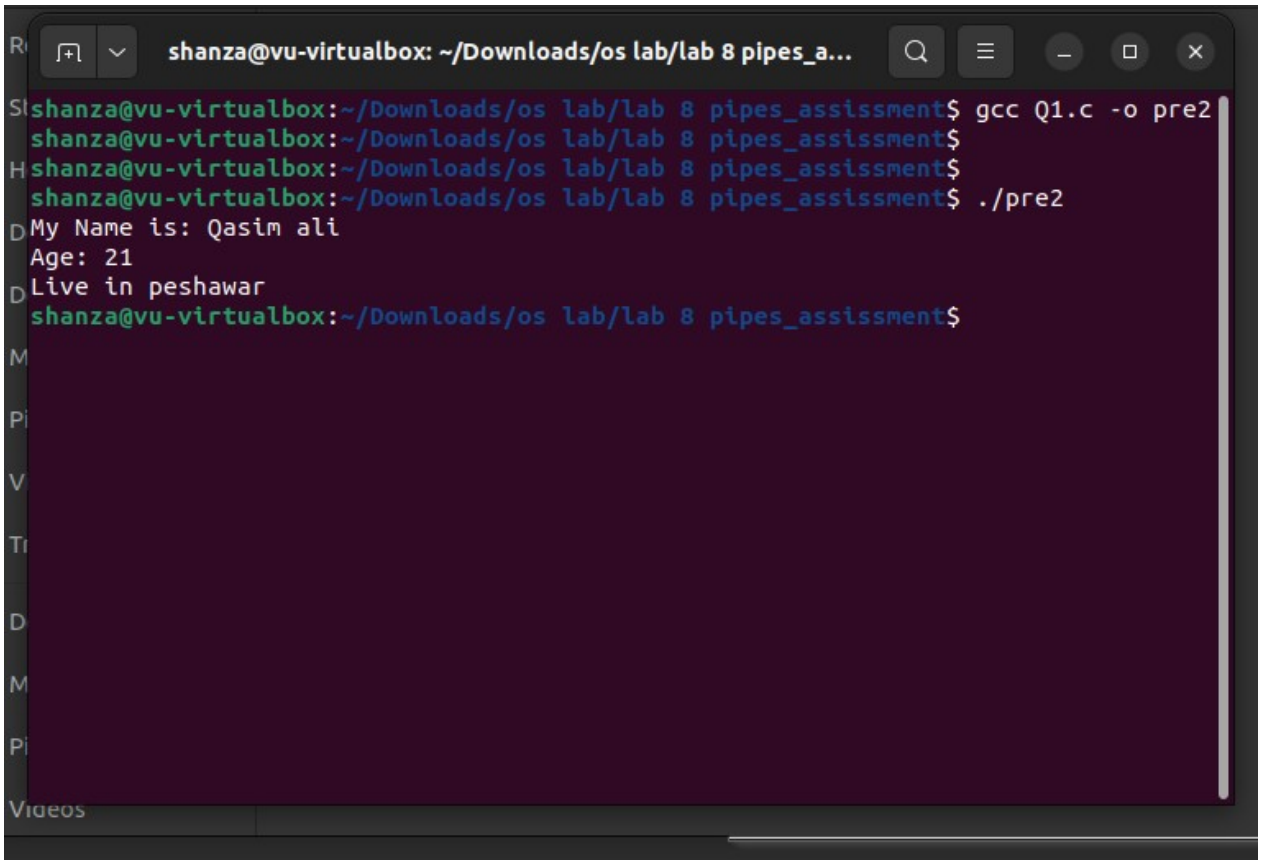
Q1: Write your name and personal information in file and then display it on output using read and write commands.

Ans:

```
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <stdio.h>
#include <signal.h>



int main() {
     // Write personal information to a file
  FILE *fp = fopen("text.c", "w");
  if (!fp) {
    printf("Error opening file for writing\n");
    return 1;
  }

  fprintf(fp, "My Name is: Qasim ali\n");
  fprintf(fp, "Age: 21\n");
  fprintf(fp, "Live in peshawar\n");

  fclose(fp);

  //Read personal information from the file
  fp = fopen("text.c", "r");
  if (!fp) {
    printf("Error opening file for reading\n");
    return 1;
  }

  char line[100];
  while (fgets(line, sizeof(line), fp)) {
    printf("%s", line);
  }
```

```
  fclose(fp);
    return 0;
}
```

the output is :



Q2: Create two different processes not in the same hierarchy, write such a program in which one process communicate with other using named pipes via read and write system calls. One process send data then wait for response message once response message received then send another message and so on. Make sure the response message will be fixed, whenever the receiver get the message it will return the fixed message in response to 1st process.

Part(A):

```
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <string.h>

int main() {
    int pipefd[2];
    char message[100];
    char response[100];

    //Create the named pipe
   int f1;
    //create a pipe with name " pipe_one " and set permissions to 6606
    f1 = mkfifo("pipe_one", 0666);



    //open the write end of the pipe
    int write_fd = open("pipe_one", O_WRONLY);

    while (20) {
        //send a message to the child process
        strcpy(message, "This is a message from the parent process.");

        write(write_fd, message, sizeof(message));

        //wait for a response message from the child process
        int read_fd = open("pipe_one", O_RDONLY);

        read(read_fd, response, sizeof(response));

        printf("Parent received response message: %s\n", response);

        close(read_fd);
```

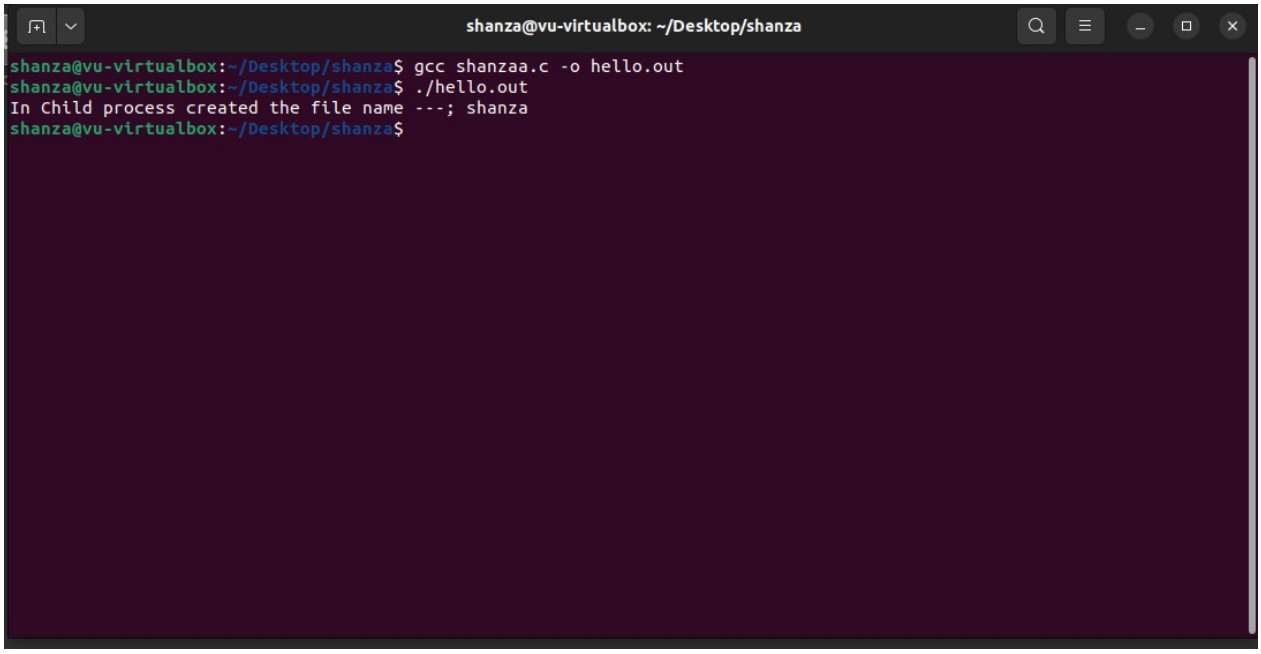
```
        sleep(1); // Optional: Add a delay between messages
    }

    close(write_fd);
    return 0;
}
```

Part(B):

```
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <string.h>

int main() {
    int pipefd[1];
    char message[100];
    char response[100];

    //open the read end of the pipe
    int read_fd = open("pipe_one", O_RDONLY);

    while (1) {
        //Wait for a message from the parent process
        read(read_fd, message, sizeof(message));

        printf("Child received message: %s\n", message);

        //Respond with a fixed message
        strcpy(response, "This is the fixed response message from
the child process.");
```

```
        int write_fd = open("pipe_one", O_WRONLY);

        write(write_fd, response, sizeof(response));
        close(write_fd);

    }

    close(read_fd);
    return 0;
}
```

The output of A and B :



```
shanza@vu-virtualbox: ~/Downloads/os lab/lab 8 pipes_assissment
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$ gcc Q2-part-b.c -o second
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$ ./second
```

After run :

Ans :

```
shanza@vu-virtualbox: ~/Downloads/os lab/lab 8 pipes_assissment
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$ gcc Q2-part-a.c -o first
Q2-part-a.c: In function 'main':
Q2-part-a.c:14:10: warning: implicit declaration of function 'mkfifo' [-Wimplicit-function-declaratio
n]
   14 |     f1 = mkfifo("pipe_one", 0666);
      |          ^~~~~~
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$ ./first
Parent received response message: This is a message from the parent process.
Parent received response message: This is a message from the parent process.
Parent received response message: This is the fixed response message from the child process.
Parent received response message: This is the fixed response message from the child process.
Parent received response message: This is the fixed response message from the child process.
Parent received response message: This is the fixed response message from the child process.
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$
```

```
shanza@vu-virtualbox: ~/Downloads/os lab/lab 8 pipes_assissment
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$ gcc Q2-part-b.c -o second
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$ ./second
Child received message: This is a message from the parent process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is a message from the parent process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is a message from the parent process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
Child received message: This is a message from the parent process.
Child received message: This is the fixed response message from the child process.
Child received message: This is the fixed response message from the child process.
^C
shanza@vu-virtualbox:~/Downloads/os lab/lab 8 pipes_assissment$
```