HRITIK BANSAL, CSE A, B1, 15, 180905105

LAB5:

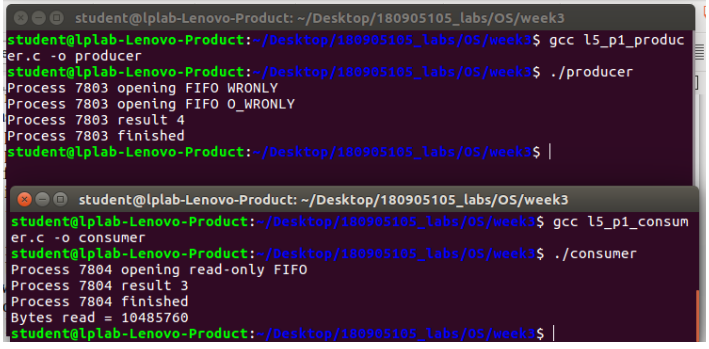
Q1) //producer

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

#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE PIPE_BUF
#define TEN_MEG (1024*1024*10)

int main(int argc, char *argv[])
{
        int pipe_fd;
   int res;
   int open_mode = O_WRONLY;
   int bytes_sent = 0;
   char buffer[BUFFER_SIZE + 1];
   if (access(FIFO_NAME, F_OK) == -1) {
      res = mkfifo(FIFO_NAME, 0777);
      if (res!=0) {
         fprintf(stderr, "Could not create fifo%s\n", FIFO_NAME);
         exit(EXIT_FAILURE);
      }
   }
   printf("Process %d opening FIFO WRONLY\n", getpid());
   pipe_fd = open(FIFO_NAME, open_mode);
   printf("Process %d opening FIFO O_WRONLY\n", getpid());
   pipe_fd = open(FIFO_NAME, open_mode);
   printf("Process %d result %d\n", getpid(), pipe_fd);
   if (pipe_fd != -1) {
      while (bytes_sent < TEN_MEG) {
         res = write(pipe_fd, buffer, BUFFER_SIZE);
         if (res == -1) {
            fprintf(stderr, "Write error on pipe\n");
            exit(EXIT_FAILURE);
         }
         bytes_sent += res;
      }
      (void)close(pipe_fd);
   }
   else {
      exit(EXIT_FAILURE);
   }
   printf("Process %d finished\n",getpid());
   exit(EXIT_SUCCESS);
}
```

// **consumer**

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

#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE PIPE_BUF

int main(int argc, char *argv[])
{
        int pipe_fd;
        int res;
        int open_mode = O_RDONLY;
   char buffer[BUFFER_SIZE + 1];
        int bytes_read = 0;
   memset(buffer,'\0',sizeof(buffer));
        printf("Process %d opening read-only FIFO\n", getpid());
        pipe_fd = open(FIFO_NAME, open_mode);
   printf("Process %d result %d\n", getpid(), pipe_fd);
        if(pipe_fd!=-1)
        {
                do {
        res = read(pipe_fd, buffer, BUFFER_SIZE);
        bytes_read += res;
      } while (res > 0);
      (void)close(pipe_fd);
        }
        else
        {
                perror("pipe");
                exit(EXIT_FAILURE);
        }
        printf("Process %d finished\nBytes read = %d\n", getpid(), bytes_read);
        exit(EXIT_SUCCESS);
        return 0;
}
```

output:

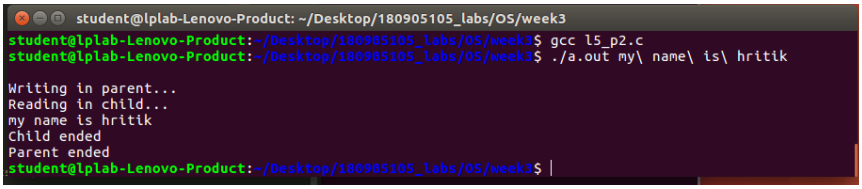


Q2)

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

int main(int argc, char *argv[])
{
	int fd[2];
	pid_t pid;
	char buf;
	if(argc!=2)
	{
		printf("Invalid no. of arguments\n");
		exit(EXIT_FAILURE);
	}
	if(pipe(fd)==-1)
	{
		perror("pipe");
		exit(EXIT_FAILURE);
	}
	pid = fork();
	if(pid==-1)
	{
		perror("fork");
		exit(EXIT_FAILURE);
	}
	else if(pid==0) //child process
	{
		printf("Reading in child...\n");
		close(fd[1]); //close unused write end
		while(read(fd[0], &buf, 1)>0)
			write(STDOUT_FILENO, &buf, 1);
		write(STDOUT_FILENO, "\n", 1);
		close(fd[0]);
		printf("Child ended\n");
		exit(EXIT_SUCCESS);
	}
	else //parent process
	{
		printf("\nWriting in parent...\n");
		close(fd[0]); //close unused read end
		write(fd[1], argv[1], strlen(argv[1]));
		close(fd[1]); //reader will see EOF
		wait(NULL); //wait for child to terminate
		printf("Parent ended\n");
		exit(EXIT_SUCCESS);
```

```
        }
        return 0;
}
```

output:



Q3)//write end

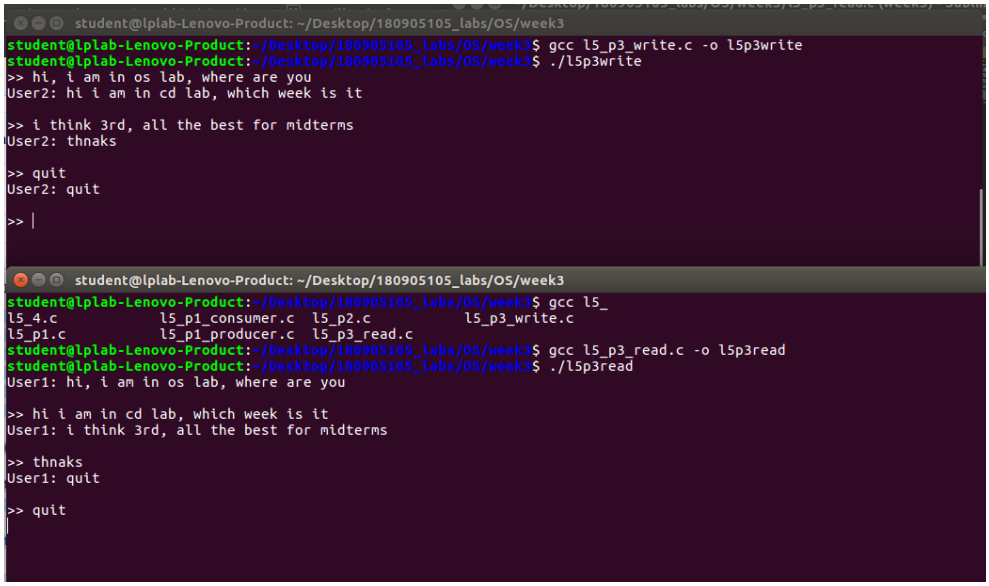
```
#include <stdio.h>
#include <string.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
    int fd;
    char * myfifo = "/tmp/myfifo";

    mkfifo(myfifo, 0777);

    char arr1[80], arr2[80];
    while (1)
    {
        fd = open(myfifo, O_WRONLY);
                    printf(">> ");
        fgets(arr2, 80, stdin);
        write(fd, arr2, strlen(arr2)+1);
        close(fd);

        fd = open(myfifo, O_RDONLY);
        read(fd, arr1, sizeof(arr1));

        printf("User2: %s\n", arr1);
        close(fd);
    }
    return 0;
}
```

//read end

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

```
int main()
{
   int fd1;

   char * myfifo = "/tmp/myfifo";
   mkfifo(myfifo, 0777);

   char str1[80], str2[80];
   while (1)
   {
      fd1 = open(myfifo,O_RDONLY);
      read(fd1, str1, 80);

      printf("User1: %s\n", str1);
      close(fd1);

      fd1 = open(myfifo,O_WRONLY);
                  printf(">> ");
      fgets(str2, 80, stdin);
      write(fd1, str2, strlen(str2)+1);
      close(fd1);
   }
   return 0;
}
```

output:



Q4)

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

int main(int argc, char** argv) {
   if (argc < 3) {
      printf("Less Arguments.\n");
```

```
        printf("Usage: %s <bin_file_name> <string_to_write_there>\n", argv[0]);
        exit(-1);
    }
    FILE* f = fopen(argv[1], "wb");
    fwrite(argv[2], strlen(argv[2]),1,f);
    close(f);
    printf("Written the file.\n\n");

    f = fopen(argv[1], "rb");
    char buff[strlen(argv[2])];
    fread(buff, strlen(argv[2]),1,f);
    for(int i = 0; i < strlen(argv[2]); ++i) printf("%x ", buff[i]);
    close(f);
    return 0;
}
```

output: