**OPERATING SYSTEM LAB 5**

**Q1.**

PROGRAM:

**Producer:**

#include<sys/types.h>

#include<sys/stat.h>

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<limits.h>

#include<fcntl.h>

#include<string.h>

#define FIFO\_NAME "/tmp/my\_fifo"

#define BUFFER\_SIZE PIPE\_BUF

#define TEN\_MEG (1024 \* 1024 \* 10)

int main(int argc, char const \*argv[])

{

int pipe\_fd;

int res;

int open\_mode = O\_WRONLY;

int bytes\_sent = 0;

int buffer;

if(access(FIFO\_NAME, F\_OK) == -1){

printf("pipe does not exist\n");

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 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 < 4\*sizeof(int)){

printf("\nEnter an integer: ");

scanf("%d", &buffer);

res = write(pipe\_fd, &buffer, sizeof(int));

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);

return 0;

}

**Consumer:**

#include<sys/types.h>

#include<sys/stat.h>

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<limits.h>

#include<fcntl.h>

#include<string.h>

#define FIFO\_NAME "/tmp/my\_fifo"

#define BUFFER\_SIZE PIPE\_BUF

int main(int argc, char const \*argv[])

{

int pipe\_fd;

int res;

int open\_mode = O\_RDONLY;

int bytes\_read = 0;

int buffer;

printf("Process %d opening FIFO O\_RDONLY\n", getpid());

pipe\_fd = open(FIFO\_NAME, open\_mode);

printf("Process %d result %d\n", getpid(), pipe\_fd);

int count = 0;

if(pipe\_fd != -1){

do{

res = read(pipe\_fd, &buffer, sizeof(int));

printf("\nInteger read is: %d, of size: %d\n", buffer, res);

bytes\_read += res;

count++;

}while(bytes\_read < 4 \* sizeof(int));

(void)close(pipe\_fd);

}

else

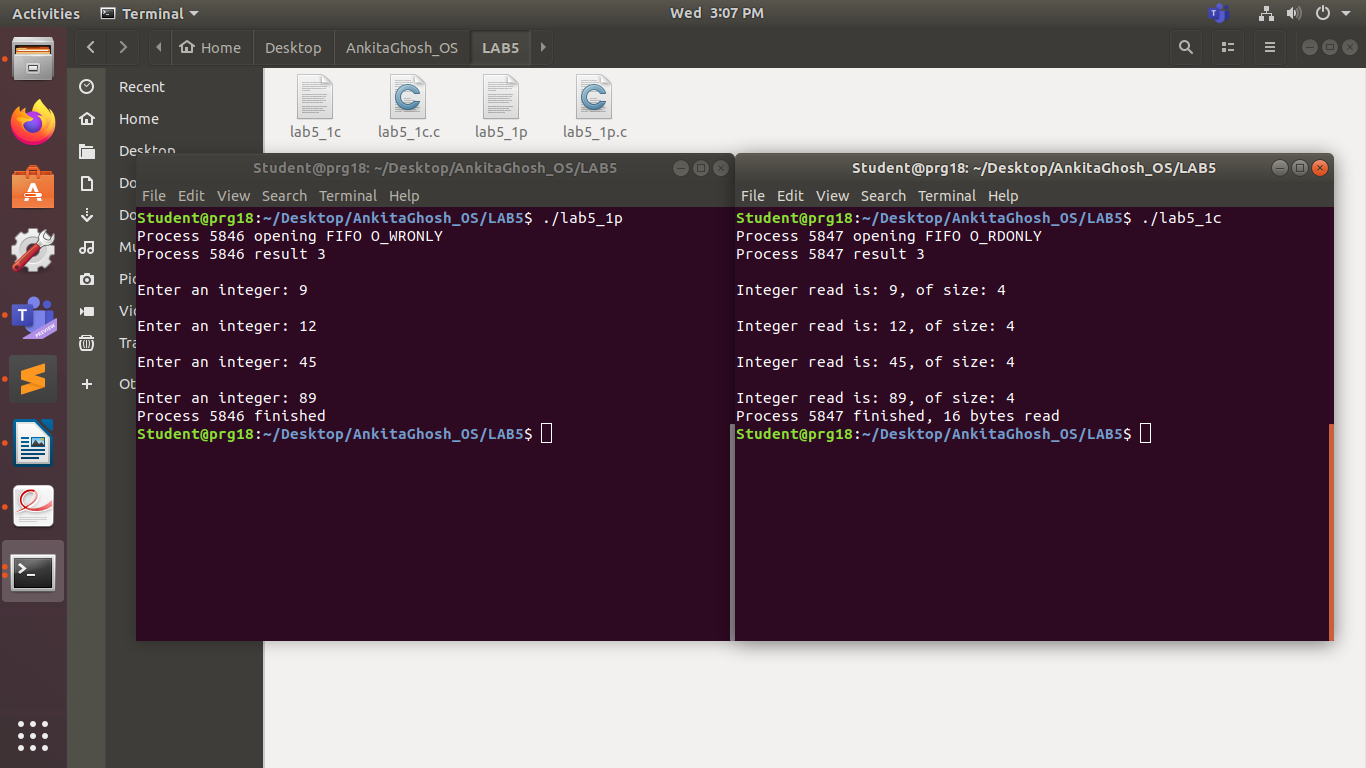
exit(EXIT\_FAILURE);

printf("Process %d finished, %d bytes read\n", getpid(), bytes\_read);

exit(EXIT\_SUCCESS);

return 0;

}  
  
OUTPUT:



**Q2.**

PROGRAM:

#include<unistd.h>

#include<stdio.h>

#include<sys/wait.h>

#include<assert.h>

#include<stdlib.h>

#include<string.h>

int main(int argc, char const \*argv[])

{

int pfd[2];

pid\_t cpid;

char buf;

if(pipe(pfd) == -1){

perror("pipe");

exit(EXIT\_FAILURE);

}

cpid = fork();

if(cpid == -1){

perror("fork");

exit(EXIT\_FAILURE);

}

if(cpid == 0){

close(pfd[0]);

int x;

printf("Enter a no: ");

scanf("%d", &x);

write(pfd[1], &x, sizeof(int));

close(pfd[1]);

exit(EXIT\_SUCCESS);

}

else

{

close(pfd[1]);

int y;

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

close(pfd[0]);

printf("From child: recieved number %d\n", y);

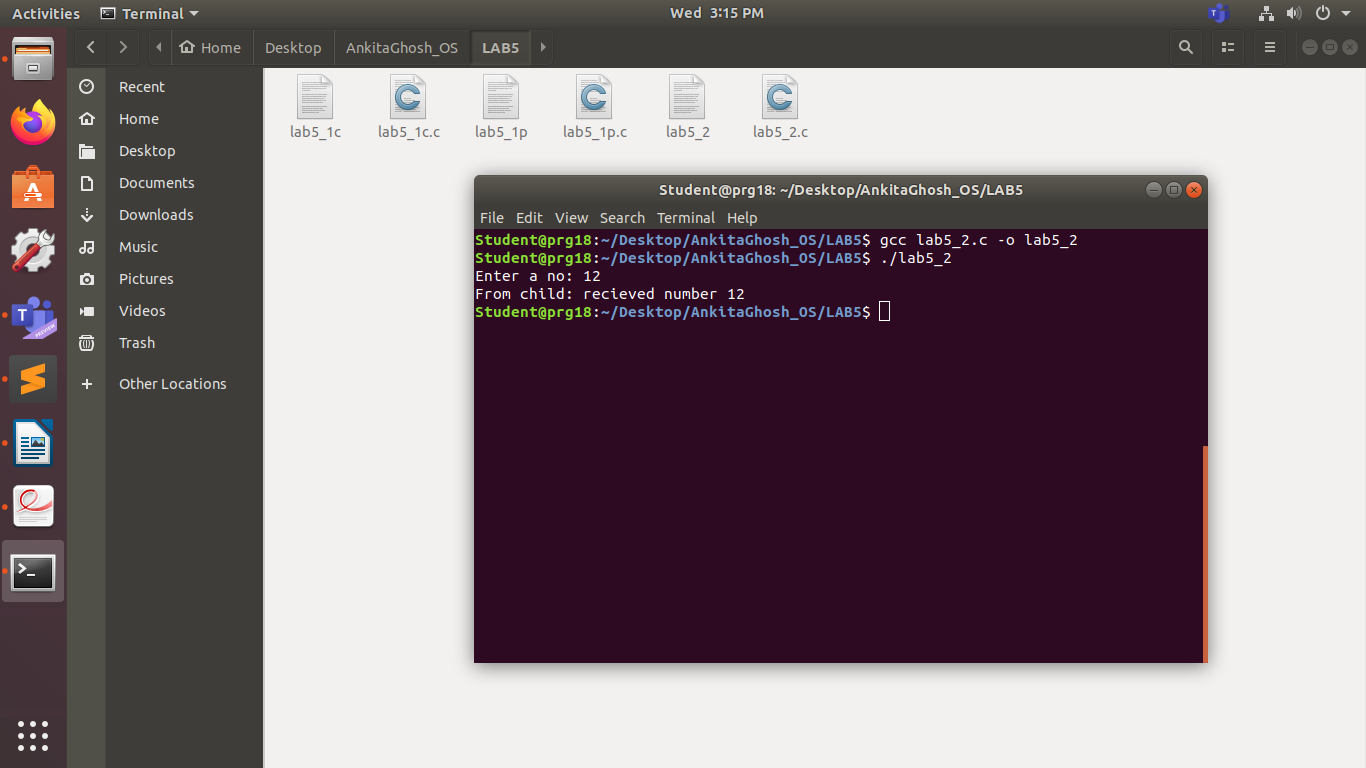
wait(NULL);

}

return 0;

}

OUTPUT:



**Q3.**

PROGRAM:

**Process 1:**

#include<sys/types.h>

#include<sys/stat.h>

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<limits.h>

#include<fcntl.h>

#include<string.h>

#define FIFO\_NAME "/tmp/my\_fifo"

#define BUFFER\_SIZE PIPE\_BUF

int main(int argc, char const \*argv[])

{

char buffer[1024];

memset(buffer, '\0', sizeof(buffer));

int res, pipe\_fd;

if(access(FIFO\_NAME, F\_OK) == -1){

printf("pipe does not exist\n");

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 O\_WRONLY\n", getpid());

pipe\_fd = open(FIFO\_NAME, O\_WRONLY);

printf("Input: ");

fgets(buffer, 1024, stdin);

write(pipe\_fd, buffer, strlen(buffer)+1);

close(pipe\_fd);

printf("\nProcess %d opening FIFO O\_RDONLY\n", getpid());

pipe\_fd = open(FIFO\_NAME, O\_RDONLY);

memset(buffer, '\0', sizeof(buffer));

read(pipe\_fd, buffer, 1024);

close(pipe\_fd);

printf("Output from Process %d: %s\n", getpid(), buffer);

return 0;

}

**Process 2:**  
#include<sys/types.h>

#include<sys/stat.h>

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<limits.h>

#include<fcntl.h>

#include<string.h>

#define FIFO\_NAME "/tmp/my\_fifo"

#define BUFFER\_SIZE PIPE\_BUF

int main(int argc, char const \*argv[])

{

char buffer[1024];

int res, pipe\_fd;

if(access(FIFO\_NAME, F\_OK) == -1){

printf("pipe does not exist\n");

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 O\_RDONLY\n", getpid());

pipe\_fd = open(FIFO\_NAME, O\_RDONLY);

memset(buffer, '\0', sizeof(buffer));

read(pipe\_fd, buffer, 1024);

close(pipe\_fd);

printf("Output from process %d: %s\n", getpid(), buffer);

printf("Process %d opening FIFO O\_WRONLY\n", getpid());

pipe\_fd = open(FIFO\_NAME, O\_WRONLY);

printf("Input: ");

memset(buffer, '\0', sizeof(buffer));

fgets(buffer, 1024, stdin);

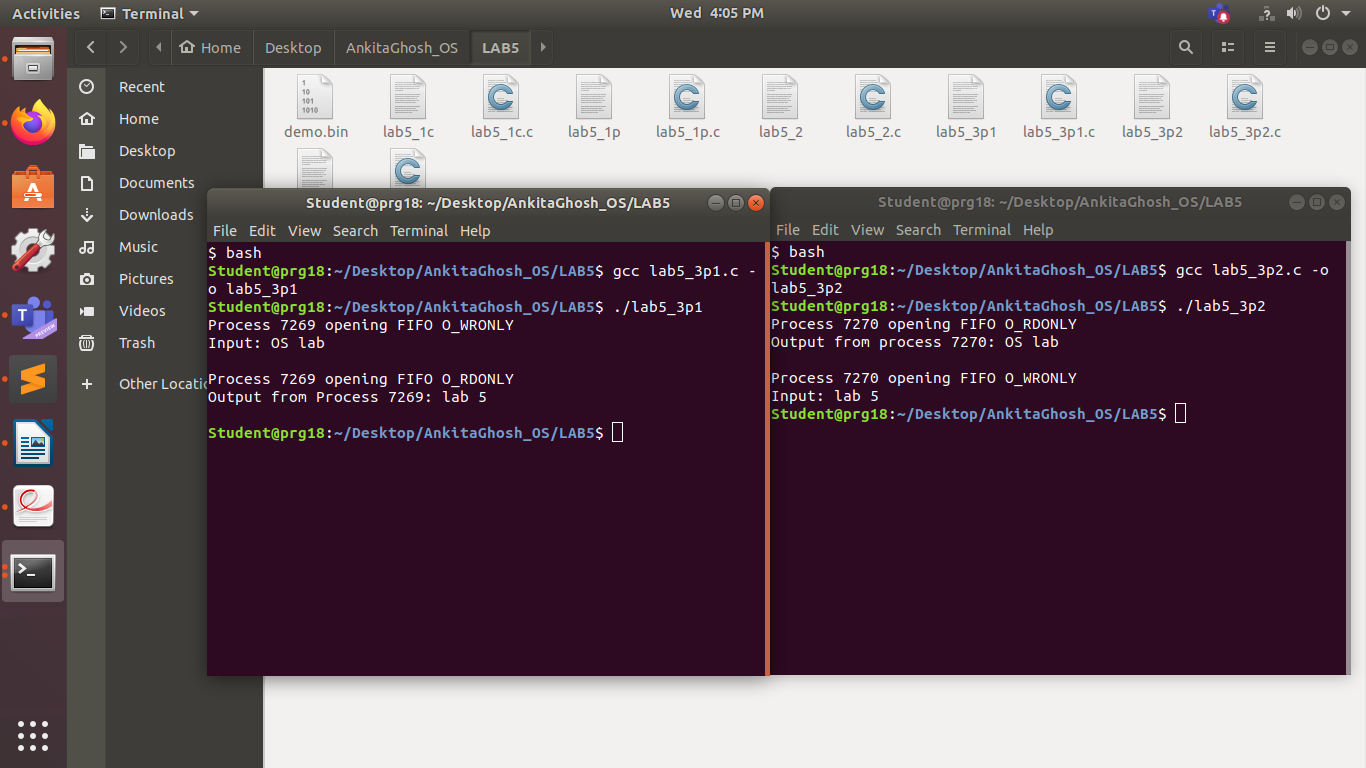
write(pipe\_fd, buffer, strlen(buffer)+1);

close(pipe\_fd);

return 0;

}

OUTPUT:



**Q4.**

PROGRAM:

#include <stdio.h>

#include <stdlib.h>

int main()

{

FILE \*fptr;

int num;

int num1=0;

char buf[50];char newbuf[50];

fptr = fopen("demo.bin","wb+");

printf("Enter input: \n");

gets(buf);

fwrite(buf, sizeof(char), 50, fptr);

printf("Write finished \n");

fclose(fptr);

fptr = fopen("demo.bin","rb");

fread(newbuf, sizeof(char), 50, fptr);

printf("%s\n",newbuf);

printf("Read finished\n");

}  
  
OUTPUT:

