OS LAB WEEK5

Name :Sagnik Chatterjee

REg :180905478

Section :B

ROll no :61

Q1.

Code:

Q1\_consumer.c

/\*

AUTHOR :SAGNIK CHATTERJEE

DATE : DEC 15,2020

USAGE : ./q1c

\*/

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

#include<fcntl.h>

#include<limits.h>

#include<sys/types.h>

#include<sys/stat.h>

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

#define BUFFER\_SIZE PIPE\_BUF

int main()

{

int pipe\_fd;

int res;

int open\_mode=O\_RDONLY;

char buffer[4];

int bytes\_read=0;

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

printf("[STATUS] Opening FIFO O\_RDONLY\n");

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

printf("[STATUS] Pipefd result :- %d \n",pipe\_fd);

if (pipe\_fd!=-1)

{

for(int i=0;i<4;i++)

{ //printing the 4 integers to the fifo queue

res=read(pipe\_fd,buffer,BUFFER\_SIZE);

if(res==-1){

printf("[ERROR] Read error on pipe.\n");

exit(1);

}

printf("%d\n",atoi(buffer));

bytes\_read+=res;

buffer[0]='\n';//clear the buffer

}

(void)close(pipe\_fd);//close the filedescriptor

}

else{

fprintf(stderr,"[ERROR] File could not be opened.\n");

exit(EXIT\_FAILURE);

}

printf("[STATUS] Finished and %d bytes read \n",bytes\_read);

exit(EXIT\_SUCCESS);

}

Q1\_producer.c

/\*

AUTHOR :SAGNIK CHATTERJEE

DATE : DEC 15,2020

USAGE : ./q1p

\*/

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

#include<fcntl.h>

#include<limits.h>

#include<sys/types.h>

#include<sys/stat.h>

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

#define BUFFER\_SIZE PIPE\_BUF

int main()

{

int pipe\_fd;

int res;

int open\_mode=O\_WRONLY;

int bytes\_sent=0;

char buffer[100];

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

res=mkfifo(FIFO\_NAME,0777);

if (res!=0)

{

fprintf(stderr,"[ERROR] Couldn't create fifo %s\n",FIFO\_NAME );

exit(EXIT\_FAILURE);

}

}

printf("[STATUS] Opening FIFO O\_WRONLY\n");

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

printf("[STATUS] Pipe\_fd result %d \n",pipe\_fd);

if (pipe\_fd!=-1)

{

for (int i=0;i<4;i++)

{ //writing the 4 integers in the fifo queue

printf(" Enter the integer \n");

scanf("%s",buffer);

res=write(pipe\_fd,buffer,100);

//buffer[0]='\n';//clear the pipe

if (res==-1)

{

fprintf(stderr,"[ERROR] Write error on pipe\n");

exit(EXIT\_FAILURE);

}

bytes\_sent+=res;

}

(void)close(pipe\_fd);//close the file descriptor

}

else

{

printf("[ERROR] Couldn't read from the pipe file descriptor.\n");

exit(EXIT\_FAILURE);

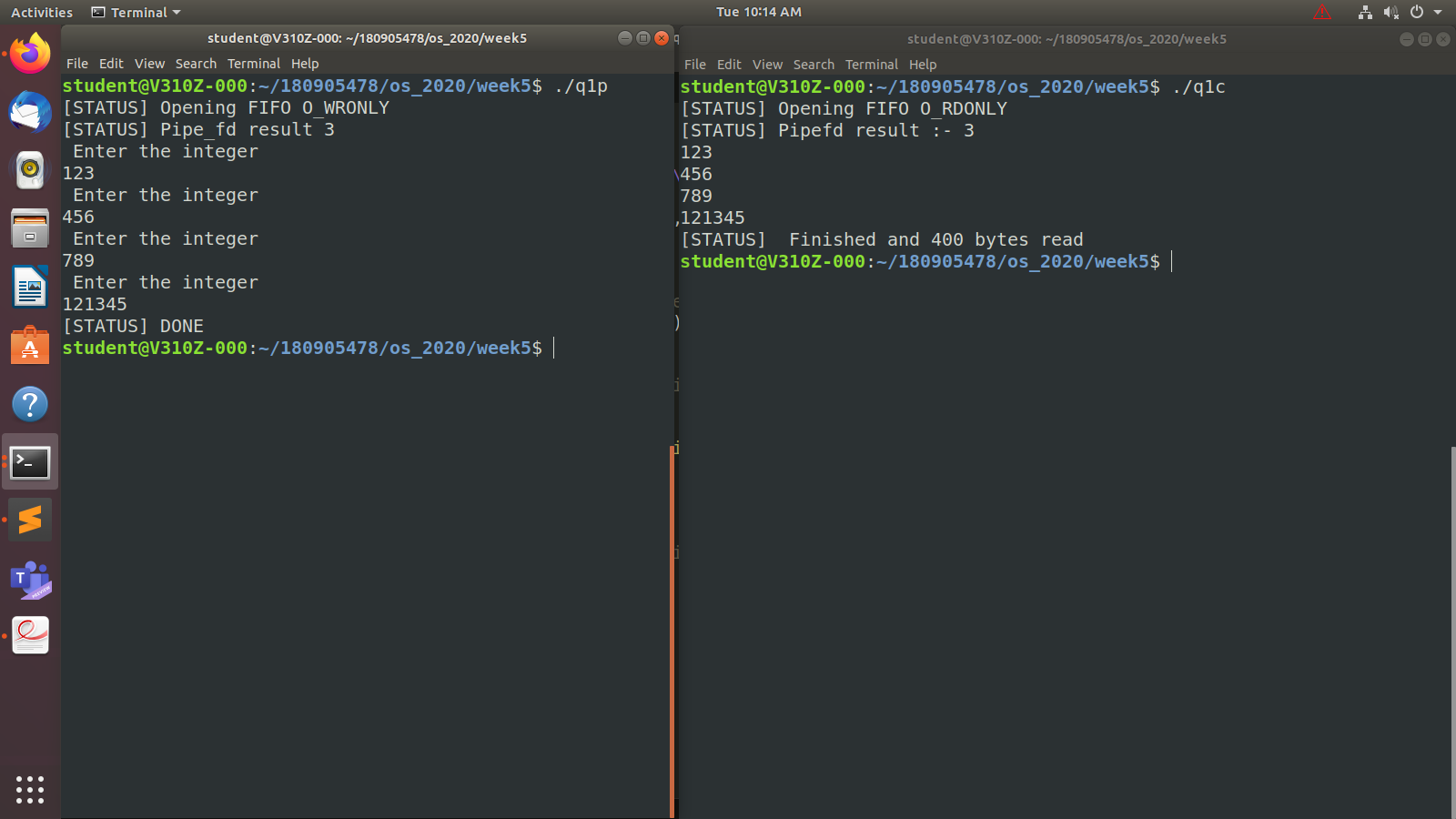
}

printf("[STATUS] DONE \n");

exit(EXIT\_SUCCESS);

}

Screenshot :



Q2

Code:

/\*

AUTHOR :SAGNIK CHATTERJEE

DATE : DEC 15,2020

USAGE : ./q2

\*/

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <limits.h>

#include <stdio.h>

int main(){

int pfd[2];

pid\_t cpid;

int buff;

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

perror("[STATUS] Pipe failure\n");

exit(EXIT\_FAILURE);

}

cpid = fork();

if(cpid==-1){

perror("[STATUS] Fork error\n");

exit(EXIT\_FAILURE);

}

else{

printf("[STATUS] Pipe created\n");

}

if(cpid==0){

//child process reads from pipe

close(pfd[1]);

int y;

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

close(pfd[0]);

printf("[STATUS] Got %d from the parent\n",y);

}

else{

//parent writes to child

close(pfd[0]);

printf("[STATUS] Enter a number\n");

scanf("%d",&buff);

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

close(pfd[1]);

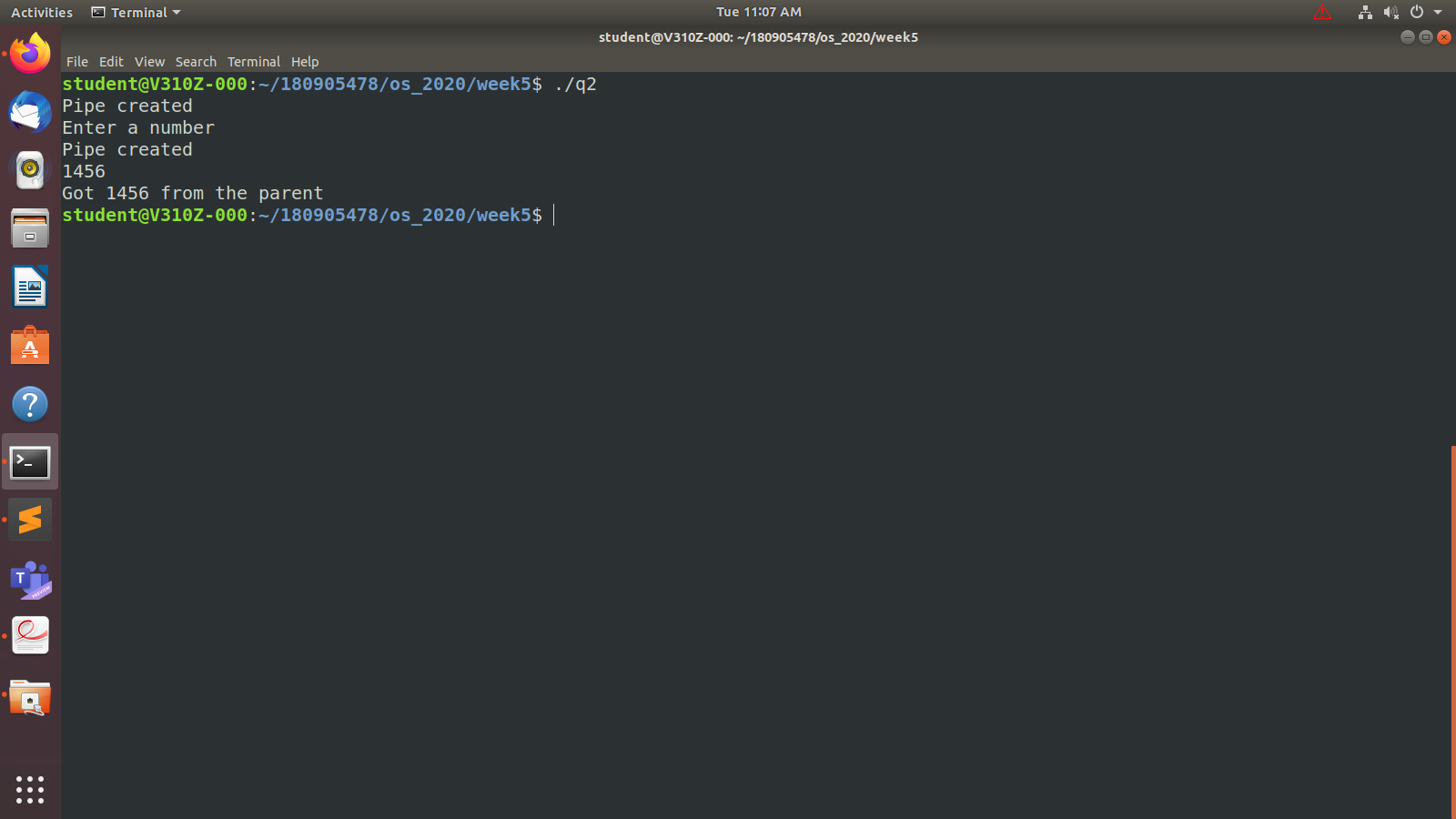
wait(NULL);

exit(EXIT\_SUCCESS);

}

}

Screenshot:



Q3

Code:

Part1 : /first writing and then reading

/\*

AUTHOR :SAGNIK CHATTERJEE

DATE : DEC 15,2020

USAGE : ./q3\_p1

\*/

#include<stdio.h>

#include<stdlib.h>

#include <string.h>

#include<fcntl.h>

#include<unistd.h>

#include<sys/wait.h>

#include<sys/stat.h>

#include<limits.h>

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

int main()

{

int pipe\_fd;

int res;

char buffer[1024];

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

//first writing and then reading

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

{

res = mkfifo(FILE\_NAME,0777);

if(res!=0)

{

fprintf(stderr,"[ERROR] Couldn't create fifo %s\n", FILE\_NAME);

exit(EXIT\_FAILURE);

}

}

printf("[STATUS] Opening FIFO\_WRONLY");

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

printf("[INPUT] Input....\n");

fgets(buffer,1024,stdin);

if(write(pipe\_fd,buffer,strlen(buffer)+1)==-1){

fprintf(stderr,"[ERROR] Error in writing \n");

exit(EXIT\_FAILURE);

}

close(pipe\_fd);

printf("[STATUS] Opening FIFO\_RDONLY\n");

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

if(pipe\_fd==-1){

fprintf(stderr,"[ERROR] Error in pie filedescriptor\n");

exit(EXIT\_FAILURE);

}

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

read(pipe\_fd,buffer,1024);

close(pipe\_fd);

printf("[STATUS] Output: Reads %s\n",buffer);

return 0;

}

Part2 : //first reading and then writing

/\*

AUTHOR :SAGNIK CHATTERJEE

DATE : DEC 15,2020

USAGE : ./q3\_p2

\*/

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <limits.h>

#include <stdio.h>

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

int main(){

int pipe\_fd;

int res;

char buffer[1024];

//first reading and then writing

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

{

res = mkfifo(FILE\_NAME,0777);

if(res!=0)

{

fprintf(stderr,"[ERROR] Couldn't create fifo %s\n", FILE\_NAME);

exit(EXIT\_FAILURE);

}

}

printf("[STATUS] Opening FIFO\_RDONLY\n");

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

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

if(read(pipe\_fd,buffer,1024)==-1){

fprintf(stderr,"[ERROR] READ error \n");

exit(EXIT\_FAILURE);

}

close(pipe\_fd);

printf("[STATUS] Output: Reads %s\n",buffer);

printf("[STATUS] Opens FIFO\_WRONLY");

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

printf("[INFO]Input....\n");

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

fgets(buffer,1024,stdin);

if(write(pipe\_fd,buffer,strlen(buffer)+1)==-1){

fprintf(stderr,"[ERROR] Error writing\n");

exit(EXIT\_FAILURE);

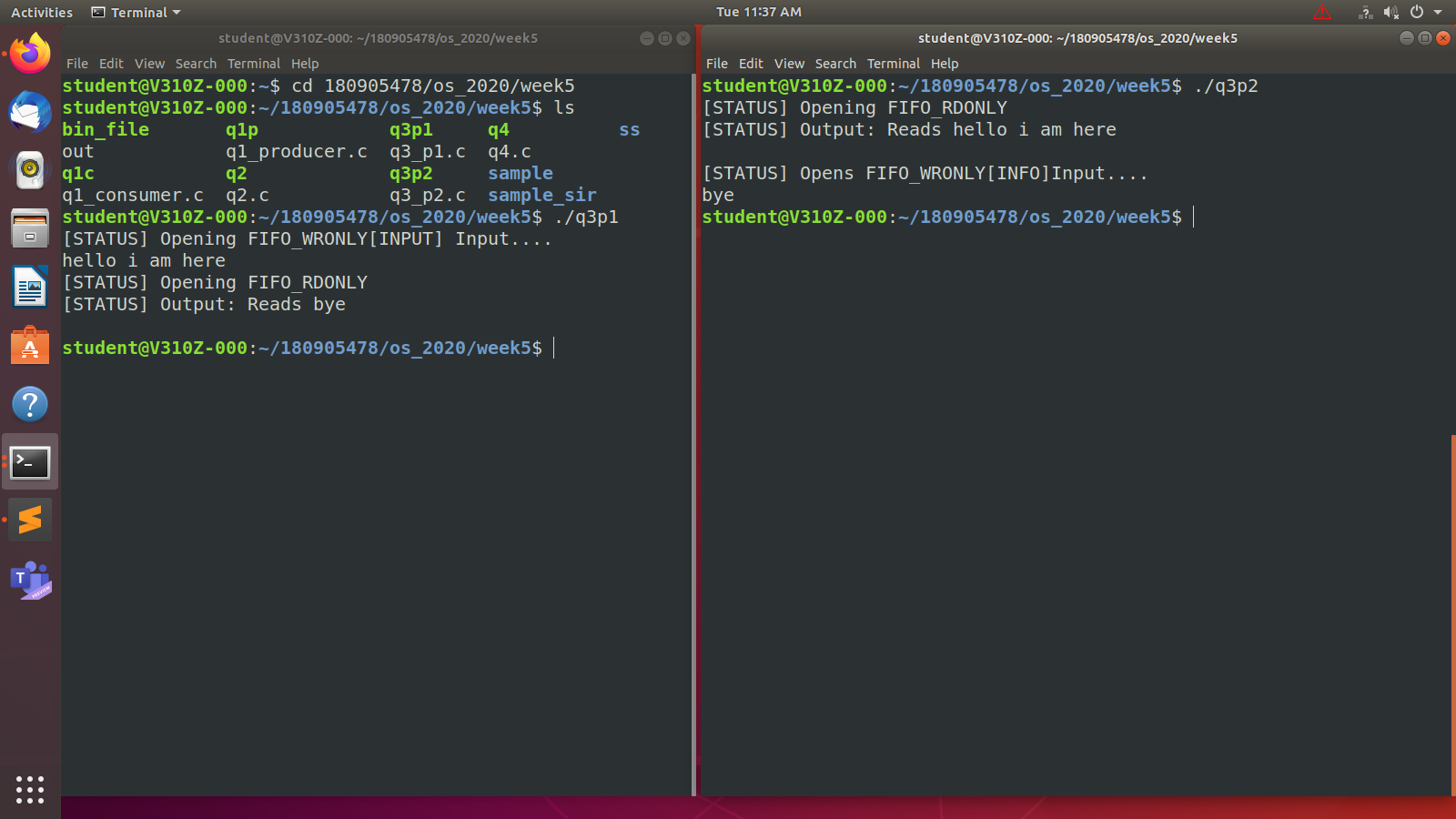
}

close(pipe\_fd);

return 0;

}

Screenshot :



Q4

Code

/\*

AUTHOR :SAGNIK CHATTERJEE

DATE : DEC 15,2020

USAGE : ./q4 inputfilename outputfilename

\*/

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/types.h>

#include <string.h>

int main(int argc , char \*\*argv) {

char buffer[1024];

FILE\* ptr;

if (argc < 3) {

printf("[ERROR] Usage : %s <filename\_to\_read> <filename\_to\_write>", argv[0]);

exit(1);

}

ptr = fopen(argv[1], "rb");

if (ptr < 0) {

printf("[ERROR] Could not open files for reading.\n");

exit(1);

}

//read the data to the buffer

if (fread(buffer, sizeof(buffer), 1, ptr) < 0) {

printf("[ERROR] Some reading error.\n");

exit(1);

}

printf("Succesfully read from the file.\n");

//print a series of bytes to screen

/\*for (int i = 0; i < 1024; i++) {

printf("%x", buffer[i]);

}

\*/

FILE\* write\_ptr ;

write\_ptr = fopen(argv[2], "wb");

if (write\_ptr < 0) {

printf("[ERROR] Could not open file for writing.\n");

exit(1);

}

if (fwrite(buffer, sizeof(buffer), 1, write\_ptr) < 0) {

printf("[ERROR] Some writing error \n");

exit(1);

}

printf("Succesfully wrote the result to file %s\n", argv[2]);

printf("\n----DONE------\n");

fclose(write\_ptr);

fclose(ptr);

return 0;

}

Screenshot:

