# Adani Institue

Laboratory Manual

DCN

()

By:

………………………Nandan Shukla……………………….

Enrolment No ...191310132139...

INFORMATION & COMMUNICATION ENGINEEING DEPARTMENT

**ADANI INSTITUTE OF INFRASTRUCTURE ENGINEERING**

Shantigram Township, Nr. Vaishnodevi Circle, SG Highway, Ahmedabad

INDEX

**Name:** Nandan Shukla\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Role No:** 191310132139\_\_\_\_\_ **Semester No:** 4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Name of Experiment** | **Marks/ Grades** | **Signature** |
| 1 | Write a C Program to implement character count Method for Client Server Programming |  |  |
| 2 | Write a C program to implement Byte Stuffing for Client Server Programming |  |  |
| 3 | Write a C program to implement Bit Stuffing for Client Server Programming |  |  |
| 4 | Write a C program to implement half duplex communication for Client Server Programming |  |  |
| 5 | Write a C program to detect error using Cyclic Redundancy Check for Client Server Programming |  |  |
| 6 | Write a C program implement classful IP addressing in networking for Client Server Programming |  |  |
| 7 | To study and test straight and cross cable using Universal Twisted Pair cable |  |  |
| 8 | Write a C program to implement shortest path routing in networking for Client Server Programming |  |  |
| 9 | Write a C program to implement stop-and-wait and sliding window protocol for Client Server Programming |  |  |
| 10 | Write a C program for TCP/IP protocol using socket programming |  |  |
| 11 | Connect Two Networks Using the Router and perform the communication among them using CISCO Packet Tracer |  |  |

**Common Instruction to write program and compile**

**To write program:**

**vi fine\_name.c**

**press i to go to insertion mode**

**after writing program, press ESC then :wq for save and quit**

**For Compilation:**

**gcc hello.c**

**For running:**

**./a.out**

**Or**

**gcc -o hello hello.c**

**now output file would be hello, so you can directly run it:**

**./hello**

**Practical 2**

**Aim:**

Write a C program to implement Byte Stuffing for Client Server Programming

1. The data transmission is to be performed using named pipes whose name is to be taken as input from command line argument.
2. Make a check in the appropriate program that the named pipe exists or not. In case the named pipe does not exist, the concerned program must create a named pipe.
3. The sender should send input of the characters from the command line argument.
4. The sender sends the data stream in a half duplex pipe.
5. The data stream is send to the receiver and receiver calculates the length of the string and sends back the length to the sender.
6. The sender gets the integer number for the length of the string and displays it on screen.

Note:

1. Assume the channel to be non – noisy.
2. The names of program, pipes and variables used should be proper and self – explanatory.
3. Follow the coding conventions of word – case where ever applicable and suitable.
4. Use appropriate data types for all the variables.

Example:

[ On sender side ] : alphanumeric

[On receiver side ] : alphanumeric

Counts the number of the characters in receiver

[ On sender side ] : character count is **12**

//Word Count program

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//server//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

#include<stdio.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<fcntl.h>

#include<string.h>

int main(void)

{

int res;

int count;

char buf[100];

res = mkfifo("myfifo",0766);

if (res == 0)

{

printf("\nFIFO created ...\n");

}

res=open("myfifo",O\_RDONLY);

read(res,buf,100);

printf("\nData is %s \n",buf);

count=strlen(buf);

printf("\nthe length of the data is %d\n",count);

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//client//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<stdio.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<fcntl.h>

#include<string.h>

int main(void)

{

int res;

char buf[100];

printf("Enter any word: ");

scanf("%s",buf);

res=open("myfifo",O\_WRONLY);

write(res,buf,100);

return 0;

}

**Output:**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//server//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**FIFO created ...**

**Data is diode**

**the length of the data is 5**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//client//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Enter any word: diode**

**Conclusion:** Hence by using this we came to know the working of the pipe and server client programming and how we can use it for different purpose like character count.

**Practical 4**

**Aim:**

Developing a C program to implement half duplex communication for Client Server Programming

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//server//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

#include<stdio.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<fcntl.h>

#include<string.h>

int main(void)

{

int p1,p2,count=0;

char buf1[100];

char buf2[100];

p1 = mkfifo("p1",0766);

p2 = mkfifo("p2",0766);

if (p1 == 0 && p2 == 0)

{

printf("\nFIFO created ...\n");

}

else

{

printf("\nFIFO is not created...\n");

}

printf("\ngive acknowledgement input\n");

scanf("%s",buf2);

for(;count<=5;)

{

p1=open("p1",O\_RDONLY);

read(p1,buf1,100);

printf("\n the data recieved is %s\n",buf1);

p2=open("p2",O\_WRONLY);

write(p2,buf2,100);

count++;

}

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//client//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<stdio.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<fcntl.h>

#include<string.h>

int main(void)

{

int p1,p2,count=0;

char buf1[100];

char buf2[100];

for(;count<=5;)

{

printf("Enter Data: ");

scanf("%s",buf1);

p1=open("p1",O\_WRONLY);

write(p1,buf1,100);

p2=open("p2",O\_RDONLY);

read(p2,buf2,100);

printf("\n%s\n",buf2);

count++;

}

return 0;

}

**Output:**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//server//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**FIFO is not created...**

**give acknowledgement input**

**DATA\_RECIEVED**

**the data recieved is Hello**

**the data recieved is I**

**the data recieved is am**

**the data recieved is diode**

**the data recieved is Nandan**

**the data recieved is Shukla**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//client//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Enter Data: Hello**

**DATA\_RECIEVED**

**Enter Data: I**

**DATA\_RECIEVED**

**Enter Data: am**

**DATA\_RECIEVED**

**Enter Data: diode**

**DATA\_RECIEVED**

**Enter Data: Nandan**

**DATA\_RECIEVED**

**Enter Data: Shukla**

**DATA\_RECIEVED**

**Conclusion:** Hence by using this we came to know the working of half duplex communication and server client programming and how we can use it.