Assignment 2

OS Lab 110698

11762 Muhammad Kashif

Task 1 (Group Chatting)

Client Code

#include<stdio.h>

#include<unistd.h>

#include<string.h>

#include<fcntl.h>

#include<sys/stat.h>

#include<sys/types.h>

#define FIFO\_FILE "MYFIFO"

int main(){

int fd2;

char\* myfifo = "/home/kali/Documents/lab\_10/myfifo";

mkfifo(myfifo,0666);

char readbuf1[80],readbuf2[80],readbuf3[80];

while(1){

printf("You can type your message\n");

fd2=open(myfifo,O\_CREAT|O\_WRONLY);

fgets(readbuf1,80,stdin);

write(fd2,readbuf1,strlen(readbuf1)+1);

close(fd2);

fd2=open(myfifo,O\_RDONLY);

read(fd2,readbuf2,sizeof(readbuf2));

printf("Read Kashif Message %s\n",readbuf2);

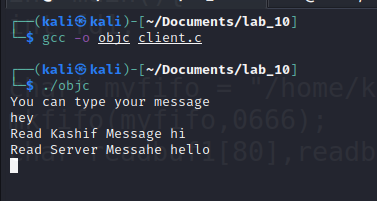
read(fd2,readbuf3,sizeof(readbuf3));

printf("Read Server Message %s\n",readbuf3);

close(fd2);}

return 0;}

Client Output



Server Code

#include<stdio.h>

#include<unistd.h>

#include<string.h>

#include<fcntl.h>

#include<sys/stat.h>

#include<sys/types.h>

#define FIFO\_FILE "MYFIFO"

int main(){

int fd;

char\* myfifo = "/home/kali/Documents/lab\_10/myfifo";

mkfifo(myfifo,0666);

char readbuf1[80],readbuf2[80],readbuf3[80];

while(1){

printf("You can type your message\n");

fd=open(myfifo,O\_CREAT|O\_WRONLY);

fgets(readbuf3,80,stdin);

write(fd,readbuf3,strlen(readbuf3)+1);

close(fd);

fd=open(myfifo,O\_RDONLY);

read(fd,readbuf1,sizeof(readbuf1));

printf("Read Client Message %s\n",readbuf1);

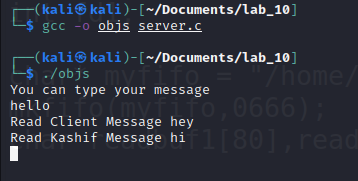
read(fd,readbuf2,sizeof(readbuf2));

printf("Read Kashif Message %s\n",readbuf2);

close(fd);}

return 0;}

Server Output



Kashif Code

#include<stdio.h>

#include<unistd.h>

#include<string.h>

#include<fcntl.h>

#include<sys/stat.h>

#include<sys/types.h>

#define FIFO\_FILE "MYFIFO"

int main(){

int fd1;

char\* myfifo = "/home/kali/Documents/lab\_10/myfifo";

mkfifo(myfifo,0666);

char readbuf1[80],readbuf2[80],readbuf3[80];

while(1){

printf("You can type your message\n");

fd1=open(myfifo,O\_CREAT|O\_WRONLY);

fgets(readbuf2,80,stdin);

write(fd1,readbuf2,strlen(readbuf2)+1);

close(fd1);

fd1=open(myfifo,O\_RDONLY);

read(fd1,readbuf1,sizeof(readbuf1));

printf("Read Client Message %s\n",readbuf1);

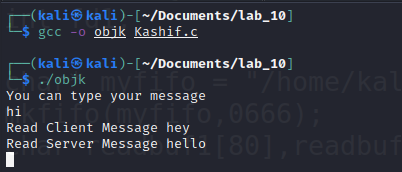
read(fd1,readbuf3,sizeof(readbuf3));

printf("Read Server Message %s\n",readbuf3);

close(fd1);}

return 0;}

Kashif Output



Task 2 (ATM Machine)

#include <stdio.h>

#include <pthread.h>

#include <unistd.h>

#define NO\_THREAD 15

#define CASH\_LIMIT 10000

#define BALANCE 150000

int cash\_balance = BALANCE;

pthread\_mutex\_t balance\_mutex;

pthread\_cond\_t client\_ready;

void \*atm\_thread(void \*arg) {

int id = \*((int \*) arg);

int password;

int choice;

int amount;

pthread\_mutex\_lock(&balance\_mutex);

pthread\_cond\_wait(&client\_ready, &balance\_mutex);

pthread\_mutex\_unlock(&balance\_mutex);

printf("Thread No %d: Please Enter your password: ", (id+1));

scanf("%d", &password);

printf("Thread No %d: Enter your choice (1 - Check Balance, 2 - Withdraw): ", (id+1));

scanf("%d", &choice);

pthread\_mutex\_lock(&balance\_mutex);

if (choice == 1) {

printf("Thread No %d: Your balance is %d\n", (id+1), cash\_balance);

} else if (choice == 2) {

printf("Thread No %d: Please Enter the amount to withdraw: ", (id+1));

scanf("%d", &amount);

if (amount > cash\_balance) {

printf("Thread No %d: Insufficient balance\n", (id+1));

} else if (cash\_balance - amount < CASH\_LIMIT) {

printf("Thread No %d: ATM is out of cash\n", (id+1));

} else {

cash\_balance -= amount;

printf("Thread No %d: Successfully withdrew %d. Your balance is %d\n", (id+1), amount, cash\_balance);}}

pthread\_mutex\_unlock(&balance\_mutex);

return NULL;}

int main(void) {

pthread\_t threads[NO\_THREAD];

pthread\_mutex\_init(&balance\_mutex, NULL);

pthread\_cond\_init(&client\_ready, NULL);

int i;

int thread\_ids[NO\_THREAD];

for (i = 0; i < NO\_THREAD; i++) {

thread\_ids[i] = i;

pthread\_create(&threads[i], NULL, atm\_thread, &thread\_ids[i]);}

for (i = 0; i < NO\_THREAD; i++) {

pthread\_cond\_signal(&client\_ready);

pthread\_join(threads[i], NULL);}

pthread\_cond\_destroy(&client\_ready);

pthread\_mutex\_destroy(&balance\_mutex);

return 0;}

Output

