PAF KIET	COLLEGE OF COMPUTING AND INFORMATION SCIENCES
	Assignment # 02

Course Title	Operating System	**Total Marks**	10
Date		**Class ID**	108185
Student Id	**11403**	**Student Name**	**Sumaiya Saleh**

Instructions:

- Copied work and late submission will be marked as ZERO.
- Attach your code and screenshot of your output in this file.
- Submit hardcopy of your solution in class.

Submission Deadline: 21-12-2021

Question 1:

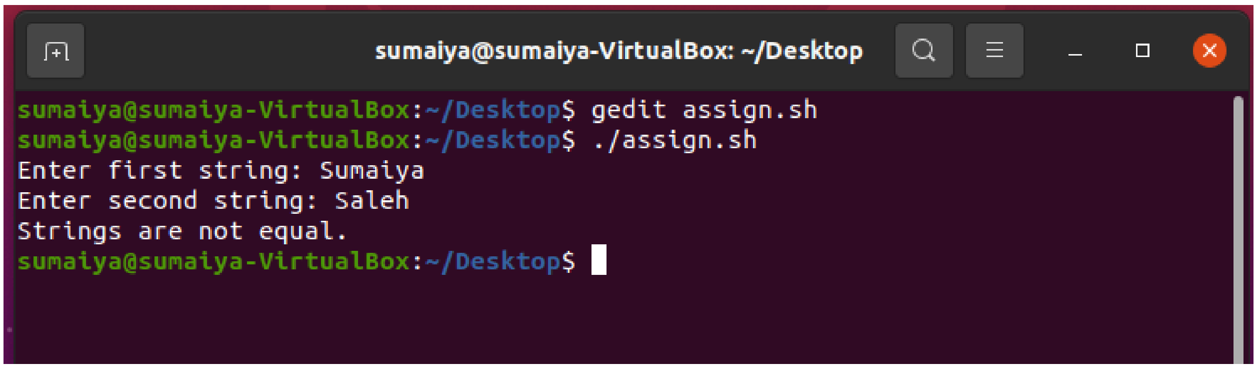
Write down the following programs using shell script:

1. Write a shell script program for comparison of strings.

Code:

```
#!/bin/bash


read -p "Enter first string: " msg1

read -p "Enter second string: " msg2


if [ "$msg1" == "$msg2" ]; then

    echo "Strings are equal."

else

    echo "Strings are not equal."

fi
```

Output:



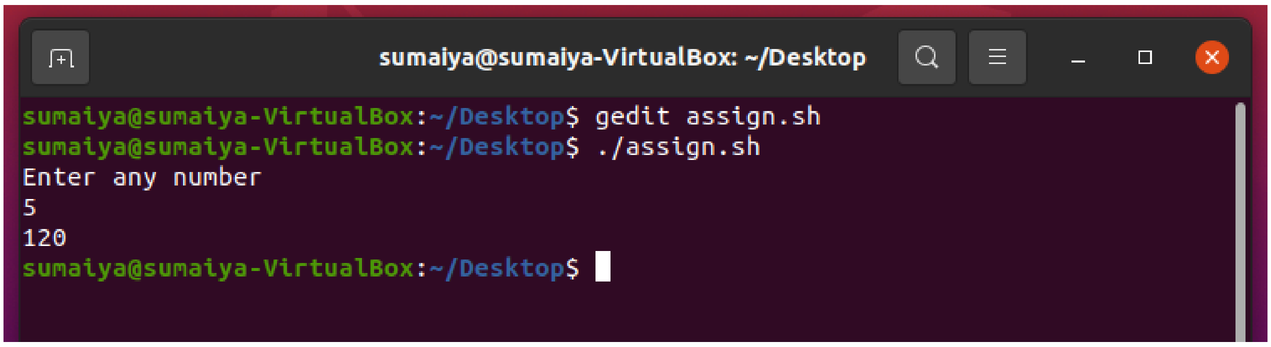
```
sumaiya@sumaiya-VirtualBox:~/Desktop$ gedit assign.sh
sumaiya@sumaiya-VirtualBox:~/Desktop$ ./assign.sh
Enter first string: Sumaiya
Enter second string: Saleh
Strings are not equal.
sumaiya@sumaiya-VirtualBox:~/Desktop$ 
```

2. Calculate the factorial value of a number using shell script.

Code:

```
#!/bin/bash
echo "Enter any number"
read number
factorial=1
for((i=2;i<=number;i++))
{
  factorial=$((factorial * i))  #factorial = factorial * i
}
echo $factorial
```

Output:



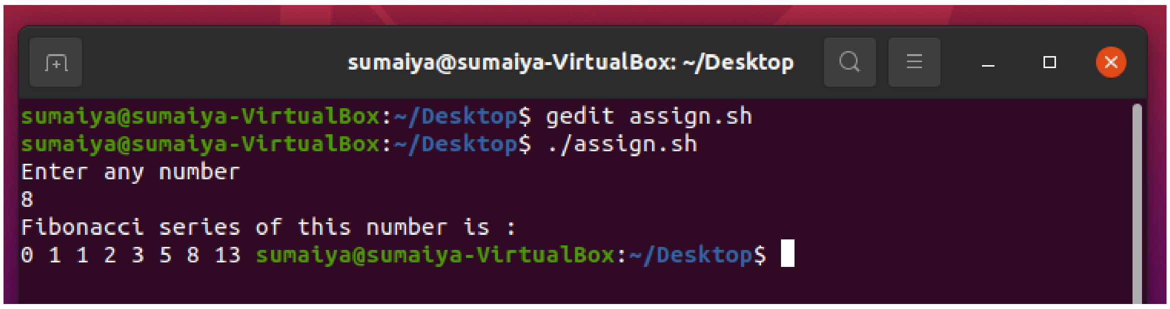
```
sumaiya@sumaiya-VirtualBox:~/Desktop$ gedit assign.sh
sumaiya@sumaiya-VirtualBox:~/Desktop$ ./assign.sh
Enter any number
5
120
sumaiya@sumaiya-VirtualBox:~/Desktop$
```

3. Write a shell program to generate Fibonacci series.

Code:

```
#!/bin/bash
echo "Enter any number"
read N
msg1=0
msg2=1
echo "Fibonacci series of this number is : "
for (( i=0; i<N; i++ ))
do
   echo -n "$msg1 "
   fn=$((msg1 + msg2))
   msg1=$msg2
   msg2=$fn
done
```

Output:



```
sumaiya@sumaiya-VirtualBox:~/Desktop$ gedit assign.sh
sumaiya@sumaiya-VirtualBox:~/Desktop$ ./assign.sh
Enter any number
8
Fibonacci series of this number is :
0 1 1 2 3 5 8 13 sumaiya@sumaiya-VirtualBox:~/Desktop$
```

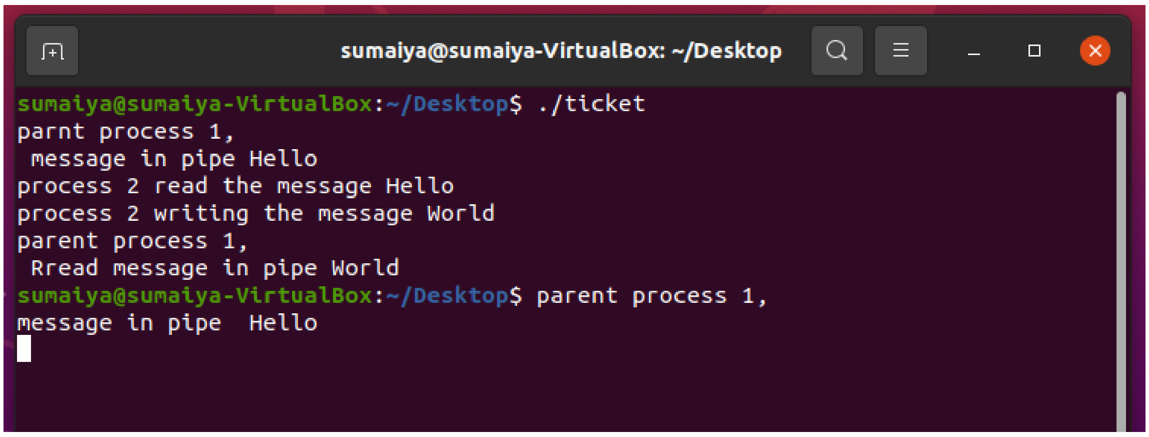
Question 2:

Think about the use of a three processes with two pipes and implement it. (You can implement any scenario of your choice).

Code:

```
#include <stdio.h>
#include <unistd.h>
int main() {
int pipe1[2],pipe2[2];
int source1;
int source2;
char pip1_m[30]="Hello";
char pip2_m[30]="World";
char read_m[30];
source1=pipe(pipe1);
if(source1==-1) {
printf("Pipe creation unsuccessfull \n\n");
return 1; }
source2=pipe(pipe2);
if(source2==-1) {
printf("Pipe creation unsuccessfull  \n\n");
}
int pid_t,child1,child2;
child1=fork();
if( child1 != 0 ){
close(pipe1[0]);
close(pipe2[1]);
printf("parnt process 1,\n message in pipe %s \n",pip1_m);
write(pipe1[1],pip1_m,sizeof(pip1_m));
read(pipe2[0], read_m,sizeof(read_m));
printf("parent process 1,\n Rread message in pipe %s              \n",read_m);
}
else{
child2=fork();
if(child2 == 0 ){
close(pipe1[0]);
close(pipe2[1]);
printf("parent process 1, \nmessage in pipe  %s  \n",pip1_m);
write(pipe1[1],pip1_m,sizeof(pip1_m));
read(pipe2[0], read_m,sizeof(read_m));
printf("parent process 1, \nread message in pipe %s        \n",read_m);
}
else{
close(pipe1[1]);
close(pipe2[0]);
read(pipe1[0], read_m, sizeof(read_m));
printf("process 2 read the message %s \n" ,read_m);
printf("process 2 writing the message %s \n",pip2_m);
write(pipe2[1],pip2_m,sizeof(pip2_m));
}}
return 0; }
```

Output:



```
sumaiya@sumaiya-VirtualBox:~/Desktop$ ./ticket
parnt process 1,
 message in pipe Hello
process 2 read the message Hello
process 2 writing the message World
parent process 1,
 Rread message in pipe World
sumaiya@sumaiya-VirtualBox:~/Desktop$ parent process 1,
message in pipe  Hello
```

Question 3:

Consider the following scenario:
There is a ticket booking counter that sells or cancels tickets for a plane seat.

- Initially, there are a total of 10 seats available numbered from 101-110. Only one person can buy or cancel a ticket at a time. A person gets the first seat available from the numbered seats.
- Every person who buys a ticket gets a ticket number and the booked seat number. The first ticket is numbered 1001 and for every successful buy, the number increases by 1.
- If a person cancels a bought ticket, that seat will be made available.
- Implement a program (write two functions **ticket_buy()** and **ticket_cancel()** to be called from main()), when there are 20 people (1 to 20) who are standing in a queue in any order to buy a ticket. 10 of these persons from the queue are initially successful in getting a ticket (ticket number

1001 to 1010). Then tickets for 3 seats are cancelled, so three next persons from the queue will get the tickets from the available seats.

- Implement while considering what happens in real life scenario when multiple people want to buy a ticket at the same time and how it is handled.

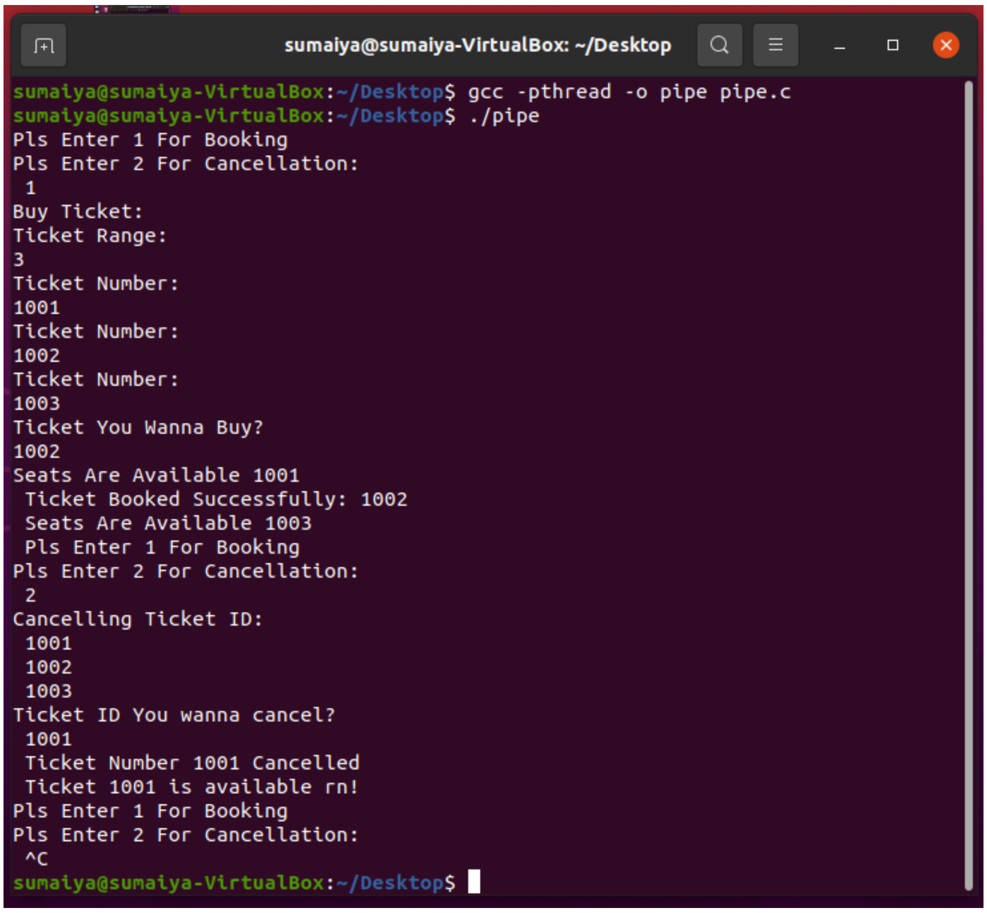
Hint: Use threads and mutex.

<u>Code:</u>

```
#include <stdio.h>
#include <unistd.h>
#include <pthread.h>
#include <stdlib.h>
pthread_mutex_t mutex1 = PTHREAD_MUTEX_INITIALIZER;
int counter = 0;
int seats[10],nmbr,i,j,passenger;
void *ticket_buy(){
pthread_mutex_lock(&mutex1);
counter++;
printf("Ticket Range: \n");
scanf("%d",&nmbr);
for(i=0; i<nmbr; i++){
printf("Ticket Number: \n");
scanf("%d",&seats[i]);
pthread_mutex_unlock(&mutex1); }
printf("Ticket You Wanna Buy? \n");
scanf("%d",&passenger);
for(i=0; i < nmbr; i++){
if(seats[i] == passenger)  {
printf("Ticket Booked Successfully: %d \n ",seats[i]); }
else{
  printf("Seats Are Available %d \n ",seats[i]);
}}}
void *ticket_cancel(){
pthread_mutex_lock(&mutex1);
counter++;
for(i=0; i<seats[i]; i++ ){
printf(" %d \n", seats[i]);
pthread_mutex_unlock(&mutex1);
}
printf("Ticket ID You wanna cancel? \n ");
scanf("%d", &seats[i]);
if(seats[i] < 0 || seats[i] > nmbr) {
        printf(" Ticket Number %d Cancelled \n", seats[i]);
```

```
    if(seats[i]==seats[i]){
        printf(" Ticket %d is available rn! \n", seats[i]);  }
    else{
      printf("Seats are not available!");
    }}}
int main (){
pthread_t t1, t2;
int r1,r2 , i ;
int selection;
for(i=0; i < 3;i++){
printf("Pls Enter 1 For Booking \nPls Enter 2 For Cancellation:\n ");
scanf("%d",&selection);
if( selection == 1){
  printf("Buy Ticket: \n");
  r1=pthread_create(&t1,NULL,ticket_buy,NULL);
  pthread_join(t1,NULL); }
else if(selection == 2){
  printf("Cancelling Ticket ID: \n");
  r2=pthread_create(&t2,NULL,ticket_cancel,NULL);
  pthread_join(t2,NULL);
}}
  return 0;
}
```

Output:



```
sumaiya@sumaiya-VirtualBox:~/Desktop$ gcc -pthread -o pipe pipe.c
sumaiya@sumaiya-VirtualBox:~/Desktop$ ./pipe
Pls Enter 1 For Booking
Pls Enter 2 For Cancellation:
 1
Buy Ticket:
Ticket Range:
3
Ticket Number:
1001
Ticket Number:
1002
Ticket Number:
1003
Ticket You Wanna Buy?
1002
Seats Are Available 1001
 Ticket Booked Successfully: 1002
 Seats Are Available 1003
 Pls Enter 1 For Booking
Pls Enter 2 For Cancellation:
 2
Cancelling Ticket ID:
 1001
 1002
 1003
Ticket ID You wanna cancel?
 1001
 Ticket Number 1001 Cancelled
 Ticket 1001 is available rn!
Pls Enter 1 For Booking
Pls Enter 2 For Cancellation:
 ^C
sumaiya@sumaiya-VirtualBox:~/Desktop$
```