CL220/205 – Operating Systems Lab

Monday, July 06, 2020

Instructor

Aliza Saeed, Saba Naseem, Muhammad Usman Joyia, Mazhar Hussain, Awais Azam

Final Term Exam
Spring Semester 2020
Max Time:3 Hour and 20 minutes
Max Marks: 75
Exam Weight (Out of 100). 40-50

18F-0128	D
Roll No	Section

Instructions:

1. The paper consists of **Objective and Subjective part.**
2. You should submit only one PDF document**. Screenshots of solution should be embedded into this word file and convert it into PDF before submission**.
3. You must submit your solution before due time via Classroom. Submissions submitted after the due time shall not be considered.
4. If you don't finish every part of a question, don't worry! You can still submit what you've done to get marks based on your efforts.
5. In case of copied or plagiarized solutions in exam Or If a student provided help to another student during exam both will be awarded "F" grade and it will affect the student CGPA.
6. Viva of any student can be conducted by the instructor after conducting an online exam in case of any doubt.
7. This document should be submitted through LMS. But in worst case, you can email it within the deadline.
8. Name your file in format "f18-XXXX_Name".

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Total
Total Marks	20	10	5	5	5	15	15	**75**
Marks Obtained								

Question no. 2 Show all your working and paste high resolution pictures of solution

Note:
-Solution should be handwritten
-Do mention your Roll No (f18-XXXX**) on top of (every page) handwritten solution**
-Then do fill the answers in the given tables too. **(10)**

Consider the following set of processes, with the length of the CPU burst given in milliseconds:

Process	Burst Time	Priority
P1	3	2
P2	2	1
P3	9	4
P4	5	2
P5	6	3

The processes are assumed to have arrived in the order **P1, P2, P3, P4, P5**, all at time **0**.

a) Draw Gantt charts that illustrate the execution of these processes using the following scheduling algorithms: **Non-preemptive priority** (a larger priority number implies a higher priority), and **RR (quantum - 2)**

b) What is the turnaround time of each process for each of the scheduling algorithms in part **a**?

	Non-preemptive Priority	**RR**
P1	**18**	**11**
P2	**25**	**4**
P3	**9**	**25**
P4	**23**	**20**
P5	**15**	**22**

c) What is the waiting time of each process for each of these scheduling algorithms?

	Non- preemptive Priority	**RR**
P1	**15**	**8**
P2	**23**	**2**
P3	**0**	**16**
P4	**18**	**15**
P5	**9**	**16**

d) Which of the algorithms results in the minimum average waiting time (over all processes)?

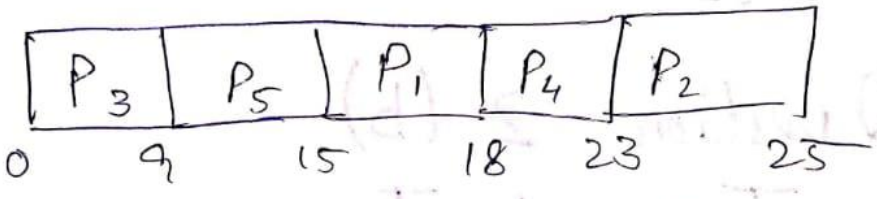
	Non- preemptive Priority	**RR**
P1	**15**	**8**
P2	**23**	**2**
P3	**16**	**0**
P4	**18**	**15**
P5	**9**	**16**

Hassan Ashas 18F-0128 CS18-D

Question 2 (a)

Non-premptive Priority

Gantt Chart :-

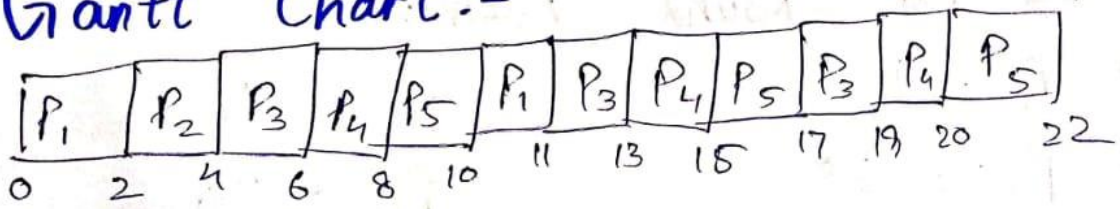


Completion Time:-

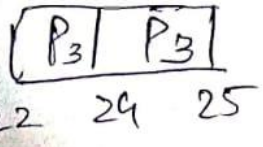
$P_3 = 9$
$P_5 = 15$
$P_1 = 18$
$P_4 = 23$
$P_2 = 25$

Round Robin

Gantt Chart :-



=>



Hasan F18-0128 CS18-D

Completion Time:-

$P_1 = 11$ $P_2 = 4$

$P_3 = 25$ $P_4 = 20$

$P_5 = 22$

Question 2 (b)

Turnaround Time

→ For non-premptive:-

• Turnaround = Completion Time - Arrival Time

$P_1 = 18$ $P_2 = 25$

$P_3 = 9$ $P_4 = 23$

$P_5 = 15$

→ For Round Robin:-

$P_1 = 11$ $P_2 = 4$

$P_3 = 25$ $P_4 = 20$

$P_5 = 22$

Hassan F18-0128 CS18-D

Question 2 (c)

Waiting Time

→ Non-premptive Priority:-

$P_1 = 15$ $P_2 = 23$

$P_3 = 0$ $P_4 = 18$

$P_5 = 9$

- Average = 13

→ Round Robin:-

$P_1 = 8$ $P_2 = 2$

$P_3 = 16$ $P_4 = 15$

$P_5 = 16$

- Average waiting time : 11.4

Question 2(d)

→ As clear from above, round robin has minimum average waiting time (11.4).

Question no. 3 **(5)**

a) Find the errors in the given shell script and fill the given table below.

```
1  # !/bin/bash
2  #Addition of three integer.
3  echo "Enter the first integer : "
4  read $fno
5  echo "Enter the second integer : "
6  read $sno
7  echo "Enter the third integer : "
8  read $tno
9  sum=expr fno + sno + tno
10 echo “The summation is: sum”
```

Errors	Description with line number
Invalid reading of fno, sno, tno variables	Line 4, 6, 8 - We don’t use $ while taking input in a variable.
Invalid use of expr	Line 9 - We must use “Backtick” with expr command for it to execute.
Accessing values of variables in expr	Line 9 - we must use $ with every variable to access its value
Not using $ with sum in echo	Line 10 – We must put $ before sum in order to extract the value of “sum” variable.

Corrected Code:

```
#!/bin/bash

echo "Enter the first integer : "
read fno
echo "Enter the second integer : "
read sno
echo "Enter the third integer : "
read tno
sum=`expr $fno + $sno + $tno`
echo "The summation is: $sum"
```

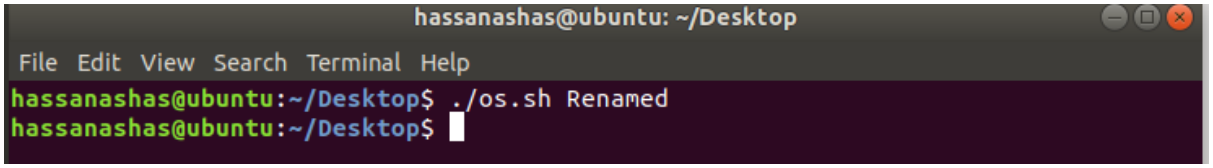
b) Explain the working of the code given below.

```
#!/bin/bash
dir=$1
for file in `ls $1/*`
do
     mv $file $file.new
done
```

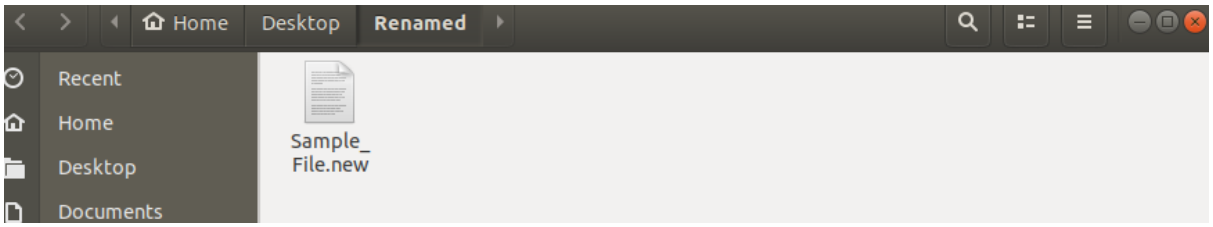
Working of Code:

- The code is basically adding the extension .new to every file to the given directory.

A directory “Renamed” is given to the code as argument (which goes as $1)



As a result, all the files in the given directory “Renamed” have an extension .new added to them. In this example, there’s only one file in the directory.



Question no. 4 **(5)**

You have been hired as an official in a firm working on a project dealing with semaphores. As a junior worker you have been assigned to work on a module whose target output must be **XAYBZCXAYBZCXAYBZC…** This module comprises of three processes namely base process, middle process and higher end process where base process prints XA, middle process prints YB and higher end process prints ZC. You are allowed to use three semaphores. Your task is to set initial value of semaphores, sequence of processes and sequence of wait() and signal() calls in each process. You are not required to write complete code.

Solution:-

- Declare three semaphores by the name of sem1, sem2, sem3
- Initial values of semaphores are set to 0 for all semaphores.
- Now, we define the three processes as follows,

Base Process:

```
Base()
{
        while(true)
        {
                wait(sem1); // waits for signal from higher end process
                cout << "XA";
                signal(sem2); // sends signal to middle process
        }
}
```

Middle Process:

```
Middle()
{
        while(true)
        {
                wait(sem2); // waits for signal from base process
                cout << "YB";
                signal(sem3); // sends signal to higher end process
        }
}
```

Higher End Process:

```
Higher_End()
{
        while(true)
        {
                wait(sem3); // waits for signal from middle process
                cout << "ZC";
                signal(sem1); // sends signal to base process (hence, starting the cycle again)
        }
}
```

Question no. 5 **(5)**

Consider two processes A and B, each accessing two semaphores C and D, set to value 1. Sequence of accessing these two semaphores is given below.

A	B
wait(C) wait(D) . . . signal(C) signal(D)	wait(D) wait(C) . . . signal(D) signal(C)

a) Show the sequence of wait and signal calls by A and B when the system enters the deadlock.

b) Show the sequence of wait and signal calls by A and B when the system does not enter any deadlock.

Solution:-

Case a:-

The **given sequence** causes deadlock situation. The process A is waiting for C, while the process B is waiting for D. Now, both processes will be waiting for the signal forever and hence, no one will move forward, causing a deadlock to occur.

Case b:-

The sequence given below will prevent the deadlock from occurring.

A	B
signal(D) wait(C) . . . signal(D) wait(C)	wait(D) signal(C) . . . wait(D) signal(C)

Question no. 6 Paste your code plus screenshots of output	(15)

NOT ATTEMPTED

Question no. 7 Paste your code plus screenshots of output	(15)

NOT ATTEMPTED

Good Luck ☺