

Assignment Report

On

Design of Operating Systems (CSE 4049)

Submitted by

Name : Saswat Mohanty

Reg. No. : 1941012407

Branch : CSE

Semester : 5th

Section : D

Session : 2021-2022

Admission Batch : 2019



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
FACULTY OF ENGINEERING & TECHNOLOGY (ITER)
SIKSHA 'O' ANUSANDHAN DEEMED TO BE UNIVERSITY
BHUBANESWAR, ODISHA – 751030**

CONTENT

Assignment No.	Name of the Assignment	Remarks
1	Process management	
2	Process scheduling	
3	Process Synchronization	
4	Deadlocks and methods of handling deadlock	
5	Memory management Strategies	
6	End term project	

Assignment #No. 1

On

Design of Operating Systems (CSE 4049)

Submitted by

Name : Saswat Mohanty

Reg. No. : 1941012407

Semester : 5th

Branch : CSE

Section : D

Session : 2021-2022

Admission Batch : 2019



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
FACULTY OF ENGINEERING & TECHNOLOGY (ITER)
SIKSHA 'O' ANUSANDHAN DEEMED TO BE UNIVERSITY
BHUBANESWAR, ODISHA – 751030**

P.S. Assignment - 1

- Q1) Write a C program to create a child process using fork() system call. The child process will print the message "child" with its process identifier and then continue in an indefinite loop. The parent process will print the message "Parent" with its process identifier and then continue in an indefinite loop.
- Run the program and trace the state of both processes.
 - Terminate the child process. Then trace the state of processes.
 - Run the program and trace the state of both processes. Terminate the parent process. Then trace the state of processes.
 - Modify the program so that the parent process after displaying the message will wait for child process to complete its task. Again run the program and trace the state of both processes.
 - Terminate the child process. Then trace the state of processes.

Program :-

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main()
{
    if (fork() == 0)
        fprintf("child %d\n", getpid());
        while (1);
    }
    else
        fprintf("Parent %d\n", getpid());
        wait(NULL);
        while (1);
}
```

return 0;
}

void ac() {

if (forkc) == 0 {
 printf ("child: %d \n", getpid());
}
else
 printf ("Parent: %d \n", getpid());
}

void bc() {

if (forkc) == 0 {
 printf ("child: %d \n", getpid());
 exit(0);
}
else
 printf ("Parent: %d \n", getpid());
}

void cc() {

if (forkc) == 0 {
 printf ("child: %d \n", getpid());
}
else {
 printf ("Parent: %d \n", getpid());
 exit(0);
}

}

void dc() {

if (forkc) == 0 {
 printf ("child: %d \n", getpid());
}
else {

(3)

```

        printf("Parent: %d \n", getpid());
        wait(NULL);
    }

}

void e() {
    if (fork() == 0) {
        printf("child: %d \n", getpid());
        _exit(0);
    }
    else {
        printf("Parent: %d \n", getpid());
        wait(NULL);
    }
}

int main() {
    a();
    b();
    c();
    d();
    e();
    return 0;
}

```

Output :-

- Parent: 29171
child: 29172
- Parent: 29218
child: 29219

c) Parent : 29491

child : 29492

d) Parent : 29640

child : 29641

e) Parent : 29772

child : 29773

Q2) Trace the output of the following program segment :

a) int main()

```
{
    if (fork() == 0)
        fprintf("1");
    else
        fprintf("2");
        fprintf("3");
    return 0;
}
```

Output :- 2313

b) int main()

```
{
    if (fork() == 0)
        fprintf("1");
    else
        fprintf("2");
        fprintf("3");
}
```

Output :- 1323

timeout : the monitored command dumped core.

a) int main()

```

{
    fid_t fid;
    int i = 5;
    fid = fork();
    i = i + 1;
    if (fid == 0)
    {
        fprintf ("child: %d", i);
    }
    else
    {
        wait (NULL);
        fprintf ("Parent: %d", i);
    }
    return 0;
}

```

Output :- child: 6
Parent: 6

d) int main()

```

{
    fid_t fid;
    int i = 5;
    fid = vfork();
    i = i + 1;
    if (fid == 0)
    {
        fprintf ("child: %d", i);
        _exit(0);
    }
    else
    {
        fprintf ("Parent: %d", i);
    }
    return 0;
}

```

Output :- child : 6
Parent : 7

e) int main()

```
{
    pid_t pid;
    int i = 5;
    pid = fork();
    if (pid == 0)
        {
            i = i + 1;
            printf("child: %d", i);
        }
    else
        {
            wait(NULL);
            printf("Parent: %d", i);
        }
    return 0;
}
```

Output :- child : 6
Parent : 5

f) int main()

```
{
    pid_t pid;
    int i = 5;
    pid = vfork();
    if (pid == 0)
        {
            i = i + 1;
            printf("child: %d", i);
            _exit(0);
        }
}
```

```

    else {
        fprintf("Parent: %d", i);
    }
    return 0;
}

```

Output :- child : 6
Parent: 6

g) int main()

```

{
    int i = 5;
    if (fork() == 0)
    {
        fprintf("child: %d", i);
    }
    else
    {
        fprintf("Parent: %d", i);
    }
    return 0;
}

```

Output :- Parent: 5
child: 5

h) int main()

```

{
    int i = 5;
    if (vfork() == 0)
    {
        fprintf("child: %d", i);
        _exit(0);
    }
    else
    {
        fprintf("Parent: %d", i);
    }
    return 0;
}

```

Output :- child: 5
Parent: 5

i) int main()

```
{
    if (fork() == 0)
    {
        fprintf("1");
    }
    else
    {
        wait(NULL);
        fprintf("2");
        fprintf("3");
    }
    return 0;
}
```

Output :- 1 2 3

j) int main()

```
{
    if (fork() == 0)
    {
        fprintf("1");
        exit(0);
    }
    else
    {
        fprintf("2");
        fprintf("3");
    }
    return 0;
}
```

Output :- 1 2 3

k) int main()

```
{
    pid_t c1;
    int n = 10;
    c1 = fork();
}
```

```
if (cl == 0)
```

{

```
printf("child \n");
```

```
n = 20
```

```
printf("n = %d \n", n);
```

{

else

{

```
mail(NULL);
```

```
printf("Parent \n");
```

```
printf("n = %d \n", n);
```

{

```
return 0;
```

{

Output :-

child	
n = 20	
Parent	
n = 10	

2) int main()

{

```
fid - t cl;
```

```
int n = 10;
```

```
cl = vfork();
```

```
If (cl == 0)
```

{

```
printf("child \n");
```

```
n = 20;
```

```
printf("n = %d \n", n);
```

```
- exit(0);
```

{

else

{

```
printf("Parent \n");
```

```
printf("n = %d \n", n);
```

{

```
return 0;
```

{

Output :-

child	
n = 20	

Parent	
n = 20	

m) int main()

```

{
    int i = 5;
    fork();
    i = i + 1;
    fork();
    fprintf(stderr, "%d", i);
    return 0;
}

```

Output :- 6 6 6 6

n) int main()

```

{
    pid_t pid;

```

```

    int l = 5;

```

```

    pid = vfork();

```

```

    if (pid == 0)

```

```

{

```

```

    printf("child: %d", l);

```

```

    _exit(0);
}

```

```

}

```

```

else
{

```

```

    l = l + 1;

```

```

    printf("Parent: %d", l);
}

```

```

return 0;
}

```

Output :- child : 5

Parent : 5

o) int main()

```

{
    int i = 5;
    if (fork() == 0)
        l = i + 1;
}

```

```

else
    i = i - 1;
    fprintf(stderr, "%d", i);
    return 0;
}

```

b) int main()

```

{
    int i = 5;
    if (fork() == 0)
    {
        i = i + 1;                                output :- 5
        exit(0);
    }
    else
        i = i + 1;
    fprintf(stderr, "%d", i);
    return 0;
}

```

c) int main()

```

{
    int j, i = 5;
    for(j = 1; j < 3; j++)
    {
        if (fork() == 0)                          output :- 6 6 5
            i = i + 1;

        break;
    }
    else
        wait(NULL);

    fprintf(stderr, "%d", i); return 0;
}

```

2) int main()

```
{  
    int j, i = 5;  
    for (j = 1; j < 3; j++)  
    {  
        if (fork() != 0)  
            i = i + 1;  
        break;  
    }  
    fprintf(stderr, "%d", i);  
    return 0;  
}
```

output :- 4 4 5

3) int main()

```
{  
    if (fork() == 0)  
        if (fork())  
            printf("1\n");  
        return 0;  
}
```

output :- 1

4) void fun1()

```
fun1();  
fun1();  
printf("1\n");
```

output :- 1
1
1
1
1
1
1
1

```
{  
end main();
```

```
fun1();  
printf("1\n");  
return 0;
```

(13)

a3) Trace the following program segment & determine how many processes are created. draw a graph that shows how the processes are related.

a) int main()

{

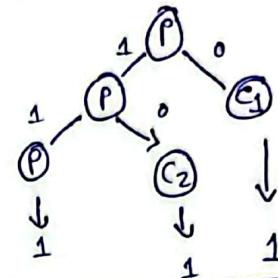
```
if(fork() && fork());
```

```
freopen("1");
```

```
return 0;
```

}

output :- 1 1 1



$$\text{Point}(111) \Rightarrow O/P = \boxed{111}$$

b) int main()

{

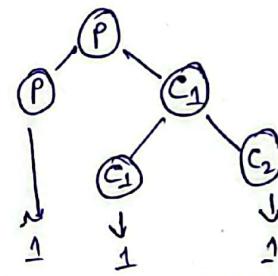
```
if(fork() || fork());
```

```
freopen("1");
```

```
return 0;
```

}

output :- 1 1 1



$$\text{Point}(111) \Rightarrow O/P = \boxed{111}$$

c) int main()

{

```
pid = fork();
```

```
C2 = 0;
```

```
C1 = fork();
```

```
if(C1 == 0)
```

```
C2 = fork();
```

```
if(C2 > 0)
```

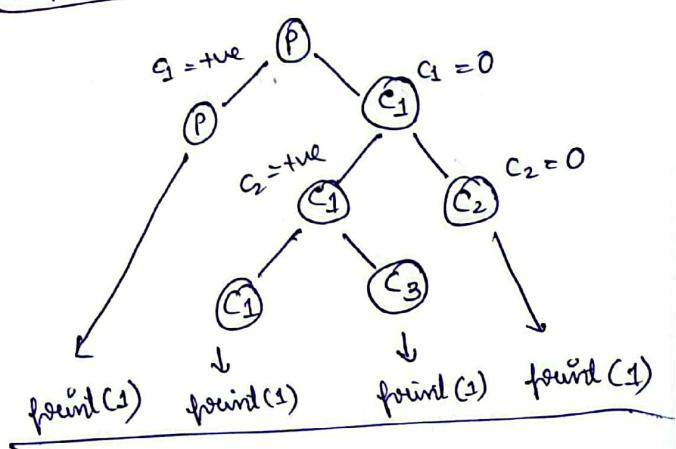
```
fork();
```

```
freopen("1");
```

```
return 0;
```

}

output :- 1 1 1 1

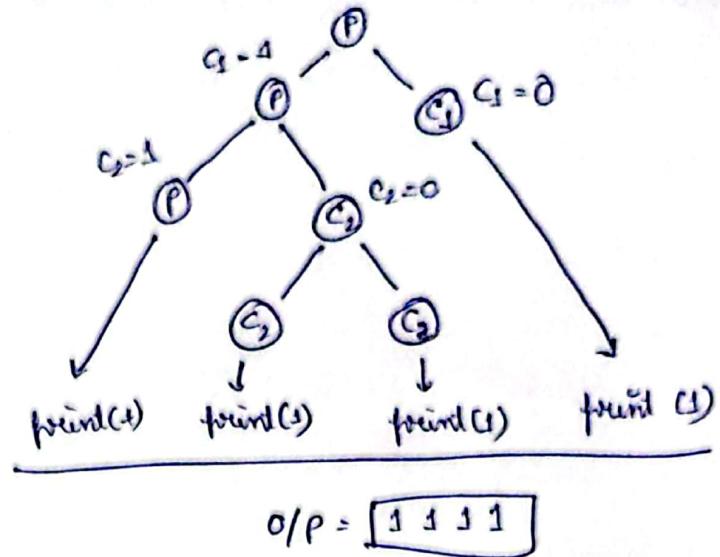


$$O/P = \boxed{1111}$$

d) int main()

```
{  
    pid_t c1 = fork();  
    c1 = fork();  
    if (c1 != 0)  
        c2 = fork();  
    if (c2 == 0)  
        fork();  
    cout << "1";  
    return 0;  
}
```

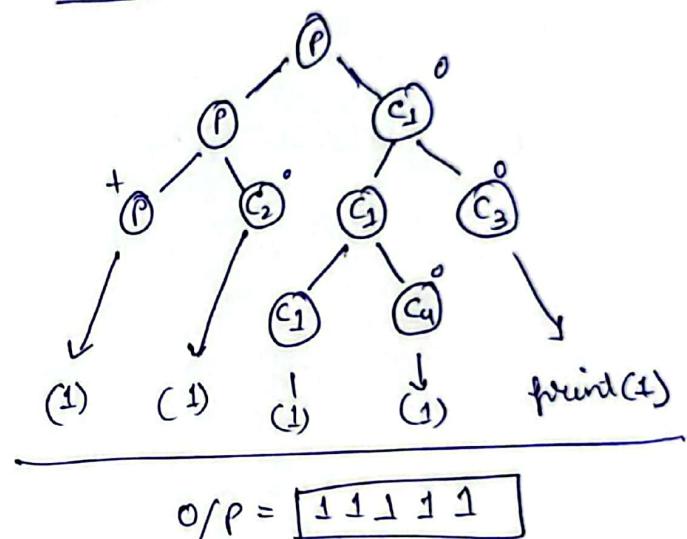
Output :- 1 1 1 1



e) int main()

```
{  
    if (fork() || fork())  
        fork();  
    cout << "1";  
    return 0;  
}
```

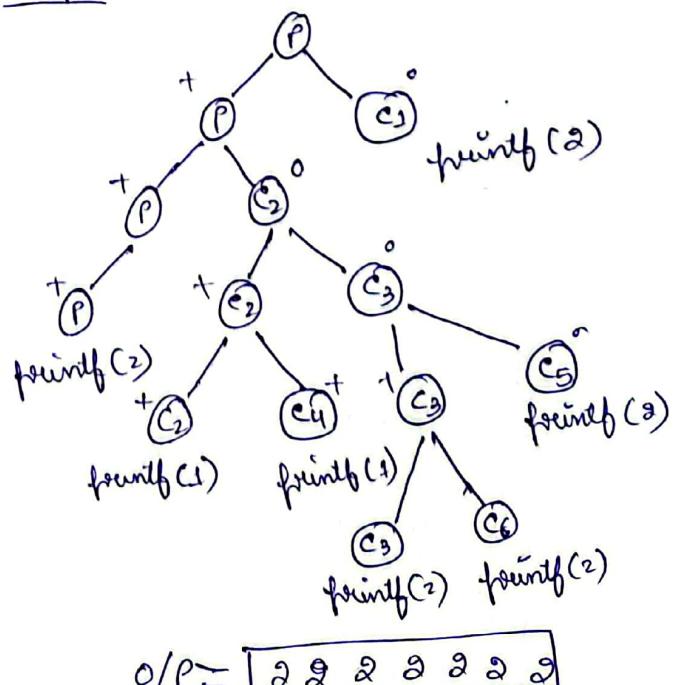
Output :- 1 1 1 1 1



f) int main()

```
{  
    if (fork() && (!fork()))  
    {  
        if (fork() || fork())  
            fork();  
    }  
    cout << "2";  
    return 0;  
}
```

Output :- 2 2 2 2 2 2



Assignment #No. 2

On

Design of Operating Systems (CSE 4049)

Submitted by

Name : Saswat Mohanty

Reg. No. : 1941012407

Semester : 5th

Branch : CSE

Section : D

Session : 2021-2022

Admission Batch : 2019



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

FACULTY OF ENGINEERING & TECHNOLOGY (ITER)

SIKSHA 'O' ANUSANDHAN DEEMED TO BE UNIVERSITY

BHUBANESWAR, ODISHA – 751030

(1)

P.S. Assignment - 2

Q1) consider the set of processes with arrival time (in milliseconds), CPU burst time (in milliseconds), and priority (low number implies high priority) as shown below:

<u>Process</u>	<u>Arrival time</u>	<u>Burst time</u>	<u>Priority</u>
P ₁	0	10	3
P ₂	0	1	1
P ₃	0	2	3
P ₄	0	1	4
P ₅	0	5	2

a) find the average turnaround time, average waiting time and average response time for each of the following scheduling algorithms along with their Gantt chart illustrating the execution of these processes:

- > FCFS, (consider the order specified in the table)
- > SJF (Non preemptive)
- > SRTF
- > Non preemptive, Priority based scheduling,
- > Preemptive Priority based scheduling
- > Round robin scheduling (quantum = 4 ms)

b) which of the algorithms result in the minimum average waiting time over all processes?

(Ans) a) FCFS :-

Gantt chart →

P ₁	P ₂	P ₃	P ₄	P ₅
0	10	11	13	14

<u>Process</u>	<u>TAT</u>	<u>WT</u>	<u>RT</u>
P ₁	10	0	0
P ₂	11	10	10
P ₃	13	11	11
P ₄	14	13	13
P ₅	19	14	14

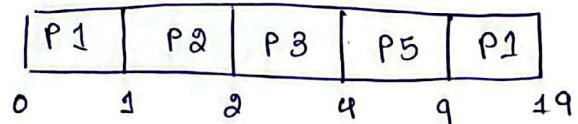
$$\text{avg. TAT} = 67/5 = 13.4 \text{ ms}$$

$$\text{avg. WT} = 48/5 = 9.6 \text{ ms}$$

$$\text{avg. RT} = 48/5 = 9.6 \text{ ms}$$

SJF (Non preemptive) :-

Gantt chart →



<u>Process</u>	<u>TAT</u>	<u>WT</u>	<u>RT</u>
P ₁	19	9	9
P ₂	1	0	0
P ₃	4	2	2
P ₄	2	1	1
P ₅	9	4	4

$$\text{avg. TAT} = 35/5 = 7 \text{ ms}$$

$$\text{avg. WT} = 16/5 = 3.2 \text{ ms}$$

$$\text{avg. RT} = 16/5 = 3.2 \text{ ms}$$

SRTF :-

Gantt chart →

P1	P2	P3	P4	P5
0	10	11	32	34

Process	WT	TAT	RT
P1	0	10	0
P2	10	11	10
P3	12	14	12
P4	11	12	11
P5	14	19	14

$$\text{avg. WT} = 47/5 = 9.4 \text{ ms}$$

$$\text{avg. TAT} = 66/5 = 13.2 \text{ ms}$$

$$\text{avg. RT} = 9.4 \text{ ms}$$

Non-preemptive, Priority based scheduling :-

Gantt chart →

P2	P5	P3	P1	P4
0	1	6	8	18

Process	TAT	WT	RT
P1	18	8	8
P2	1	0	0
P3	8	6	6
P4	19	18	18
P5	6	1	1

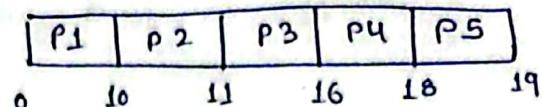
$$\text{avg. TAT} = 52/5 = 10.4 \text{ ms}$$

$$\text{avg. WT} = 33/5 = 6.6 \text{ ms}$$

$$\text{avg. RT} = 9.3/5 = 6.6 \text{ ms}$$

Preemptive, Priority based scheduling :-

gantt chart →



Process	TAT	WT	RT
P1	10	0	0
P2	11	10	10
P3	18	16	16
P4	19	18	18
P5	16	11	11

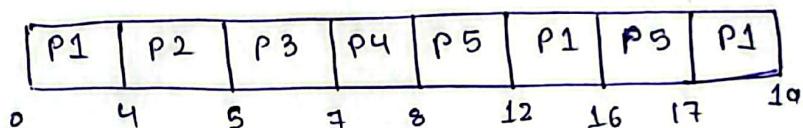
$$\text{avg. TAT} = 14.8 \text{ ms}$$

$$\text{avg. WT} = 11 \text{ ms}$$

$$\text{avg. RT} = 11 \text{ ms}$$

Round robin scheduling (quantum = 5ms) :-

gantt chart →



Process	BT	TAT	WT	RT	
P1	10	19	9	9	avg TAT = 11.2 ms
P2	10	5	4	4	avg WT = 7.4 ms
P3	10	7	5	5	avg. RT = 7.4 ms
P4	10	8	7	7	
P5	11	17	12	12	

- b) SJF has the minimum average waiting time over all processes.
- c) consider the set of processes with arrival time (in milliseconds), CPU burst time (in microseconds), and priority (low number implies high priority) as shown below.

<u>Process</u>	<u>Arrival time</u>	<u>Burst time</u>	<u>Priority</u>
P1	0	4	3
P2	0	2	1
P3	1	3	2
P4	2	2	4

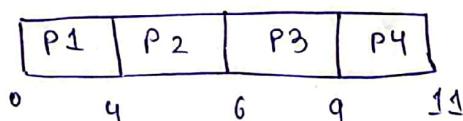
a) Find the average turnaround time, average waiting time and average response time for each of the following scheduling algorithm along with their Gantt chart illustrating the execution of these processes:

- FCFS, (consider the order specified in the table)
- SJF (Non preemptive)
- SRTF
- Non preemptive, Priority based scheduling
- Preemptive priority based scheduling.
- Round robin scheduling (quantum = 2 ms)

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

(Ans) a) FCFS :-

Gantt chart →



<u>Processes</u>	<u>TAT</u>	<u>WT</u>	<u>RT</u>
P1	4	0	0
P2	6	4	4
P3	8	5	5
P4	9	7	7

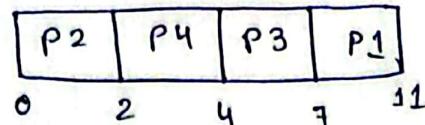
$$\text{avg. TAT} = 6.75 \text{ ms}$$

$$\text{avg. WT} = 4 \text{ ms}$$

$$\text{avg. RT} = 4 \text{ ms}$$

SJF (non-preemptive) :-

Gantt chart →



avg TAT = 5.25ms

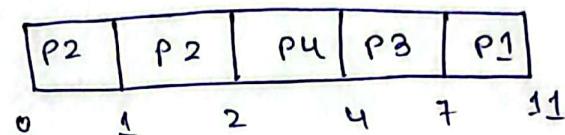
avg. WT = 2.5ms

avg. RT = 2.5ms

Process	TAT	WT	RT
P1	11	7	7
P2	2	0	0
P3	6	3	3
P4	2	0	0

SRTF :-

Gantt chart →



avg TAT = 5.25ms

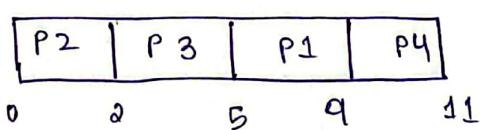
avg WT = 2.5ms

avg RT = 2.5ms

Process	BT	TAT	WT	RT
P1	4 0	11	7	7
P2	2 2 0	2	0	0
P3	2 0	6	3	3
P4	2 0	2	0	0

Non-preemptive, priority based scheduling :-

Gantt chart →



avg TAT = 6ms

avg WT = 3.25ms

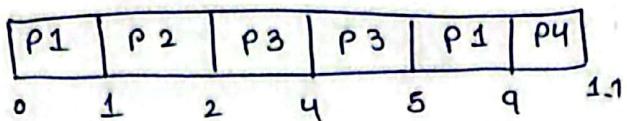
avg. RT = 3.25ms

Process	CT	TAT	WT	RT
P1	9	9	5	5
P2	2	2	0	0
P3	5	4	1	1
P4	11	9	7	7

Premptive, Priority based scheduling :-

(7)

Gantt chart →



Process	BT	CT	TAT	WT	RT
P1	4 0	9	9	5	5
P2	2 2 0	2	2	0	0
P3	3 0	5	4	1	1
P4	2 0	11	9	7	7

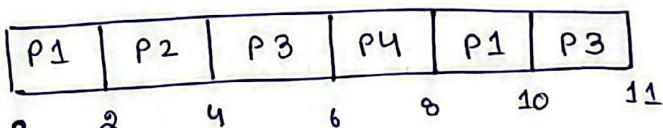
$$\text{avg TAT} = 6 \text{ ms}$$

$$\text{avg WT} = 3.25 \text{ ms}$$

$$\text{avg RT} = 3.25 \text{ ms}$$

Round robin scheduling :-

Gantt chart →



Process	BT	CT	TAT	WT	RT
P1	4 2 0	10	10	6	0
P2	2 0	4	4	2	2
P3	3 2 0	11	10	7	3
P4	2 0	8	6	4	4

$$\text{avg TAT} = 7.0 \text{ ms}$$

$$\text{avg WT} = 4.75 \text{ ms}$$

$$\text{avg RT} = 2.25 \text{ ms}$$

b) SJF & SRTF scheduling algorithms results in the minimum average waiting time over all processes.

Q3) consider the set of processes with arrival time (in milliseconds), CPU burst time (in milliseconds), and priority (high number implies high priority) as shown below :

<u>Process</u>	<u>Arrival time</u>	<u>Burst time</u>	<u>Priority</u>
P1	0	11	1
P2	0	8	0
P3	12	2	3
P4	2	6	2
P5	9	16	4

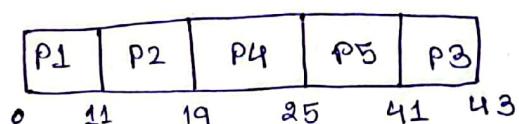
a) find the average turnaround time, average waiting time and average response time for each of the following scheduling algorithm along with their Gantt chart illustrating the execution of these processes :

- FCFS, (consider the order specified in the table)
- SJF (Non preemptive)
- SRTF
- Non preemptive , Priority based scheduling
- Preemptive , Priority based scheduling
- Round robin scheduling (quantum = 5ms)

b) which of the algorithms results in the minimum average waiting time over all process ?

(Ans) a) FCFS :-

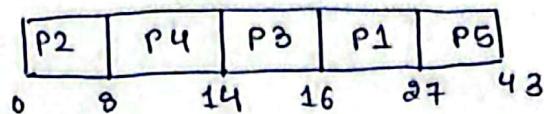
Gantt chart →



<u>Process</u>	<u>CT</u>	<u>TAT</u>	<u>WT</u>	<u>RT</u>	
P1	11	11	0	0	avg TAT = 28.2 ms
P2	19	19	11	11	avg WT = 14.6 ms
P3	43	31	29	29	avg RT = 14.6 ms
P4	25	23	17	17	
P5	41	32	16	16	

SJF (Non-preemptive) :-

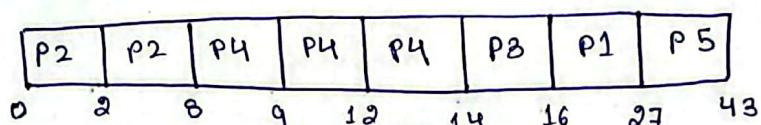
Gantt chart →



Process	CT	TAT	WT	RT	
P1	27	27	16	16	avg TAT = 17 ms
P2	8	8	0	0	avg WT = 8.4 ms
P3	16	4	2	2	avg RT = 8.4 ms
P4	14	12	6	6	
P5	43	34	18	18	

SRTF :-

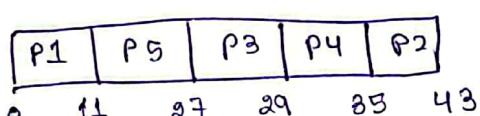
Gantt chart →



Process	BT	CT	TAT	WT	RT	
P1	11 0	27	27	16	16	avg TAT = 17 ms
P2	8 8 0	8	8	0	0	avg WT = 8.4 ms
P3	2 0	16	4	2	2	avg RT = 8.4 ms
P4	8 8 10	14	12	6	6	
P5	18 0	43	34	18	18	

Non-preemptive priority based scheduling :-

Gantt chart →

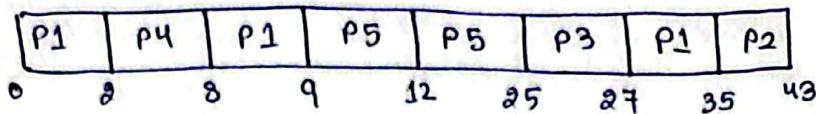


Process	CT	TAT	WT	RT	
P1	11	11	0	0	avg TAT = 24.4 ms
P2	43	43	35	35	avg WT = 15.8 ms
P3	29	17	15	15	avg RT = 15.8 ms
P4	35	33	27	27	
P5	27	19	2	2	

Premptive priority based scheduling :-

40

Gantt chart →



Process	CT	TAT	WT	RT
P1	35	35	24	0
P2	43	43	35	35
P3	27	15	13	13
P4	8	6	0	0
P5	25	16	0	0

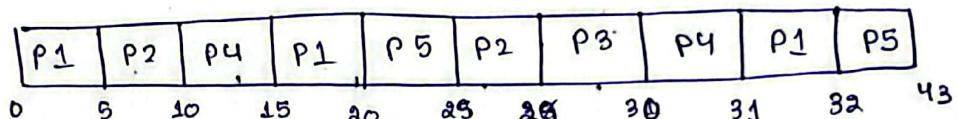
$$\text{avg TAT} = 23 \text{ ms}$$

$$\text{avg WT} = 14.4 \text{ ms}$$

$$\text{avg RT} = 9.6 \text{ ms}$$

Round Robin scheduling :-

Gantt chart →



Process	CT	TAT	WT	RT
P1	32	22	21	0
P2	20	28	20	5
P3	30	18	16	23
P4	31	29	23	10
P5	43	34	18	20

$$\text{avg TAT} = 28.2 \text{ ms}$$

$$\text{avg WT} = 19.6 \text{ ms}$$

$$\text{avg RT} = 12.6 \text{ ms}$$

- b) Both SRTF & SJF have the minimum average waiting time over all processes.

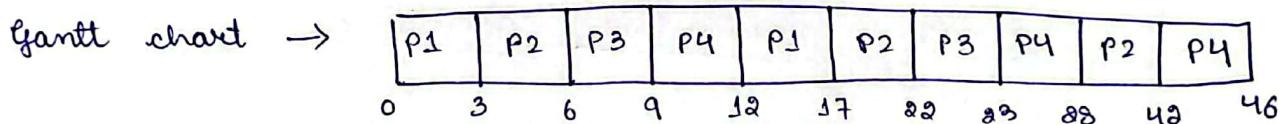
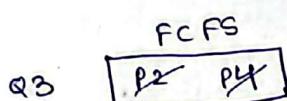
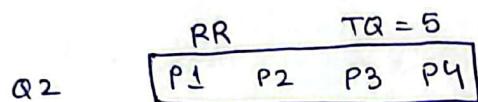
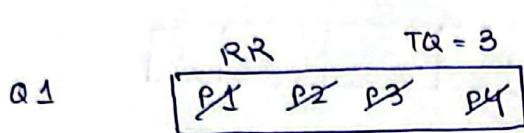
- Q4) consider a multilevel feedback queue scheduling (MLFBQ) with three queues q_1, q_2 and q_3 . q_1 and q_2 use round-robin algorithm with time quantum equal to 3 and 5 milliseconds respectively. q_3 uses first-come first-serve algorithm. Assume the arrival time of all processes as 0. A process entering the ready queue will put in queue 0. Processes in queue q_1, q_2 will be demoted to lower priority queue, if not

(11)

completed on specified time quantum. Find the average waiting time (A.W.T) and average turnaround time (A.T.A.T) for executing the following processes?

<u>Processes</u>	<u>Burst time</u>
P1	8
P2	22
P3	4
P4	12

(Ans)



<u>Process</u>	<u>AT</u>	<u>BT</u>	<u>CT</u>	<u>TAT</u>	<u>WT</u>	<u>RT</u>
P1	0	8	17	17	9	0
P2	0	22	42	42	20	3
P3	0	4	23	23	19	6
P4	0	12	46	46	34	9

$$\text{avg TAT} = 29.5 \text{ ms}$$

$$\text{avg WT} = 20.5 \text{ ms}$$

$$\text{avg RT} = 4.5 \text{ ms}$$

- Q3) Consider three CPU-intensive processes, which require 10, 20 and 30 time units and arrive at times 0, 3 and 6, respectively. How many

(12)

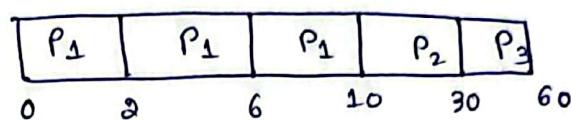
context switches are needed if the operating system implements a shortest remaining time first scheduling algorithm? Do not count the context switches at time zero and at the end..

(Ans)

<u>Process</u>	<u>BT</u>	<u>AT</u>
P ₁	10	0
P ₂	20	2
P ₃	30	6

SRTF (shortest remaining time first scheduling)

Gantt chart →



only two context switches are required i.e. P₁ to P₂ and P₂ to P₃

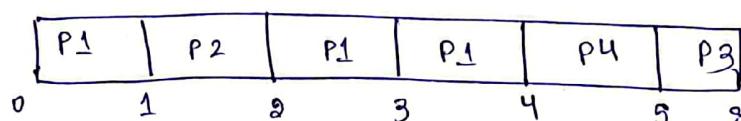
Q6) consider the following four processes with arrival times (in milliseconds) and their length of CPU bursts (in milliseconds) as shown below:

<u>Process</u>	<u>Arrival time</u>	<u>Burst time</u>
P ₁	0	3
P ₂	1	1
P ₃	3	3
P ₄	4	x

find the value of x, such that the average waiting time of the process is 1 millisecond, if the processes execute on a single processor using SRTF scheduling.

(Ans) Let x = 1

Gantt chart



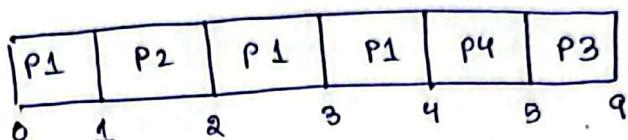
<u>Process</u>	<u>AT</u>	<u>BT</u>	<u>CT</u>	<u>TAT</u>	<u>WT</u>	<u>RT</u>
P ₁	0	3	4	4	1	0
P ₂	1	1	2	1	0	0
P ₃	3	3	8	5	2	2
P ₄	4	x=1	5	1	0	0

$$\text{avg} = \frac{3}{4}$$

$$= 0.75 \text{ ms}$$

Let x = 2

Gantt chart \rightarrow



<u>Process</u>	<u>AT</u>	<u>BT</u>	<u>CT</u>	<u>TAT</u>	<u>WT</u>	<u>RT</u>
P ₁	0	3	4	4	1	0
P ₂	1	1	2	1	0	0
P ₃	3	3	9	6	3	3
P ₄	4	x=2	6	2	0	0

$$\text{avg} = \frac{4}{4}$$

$$= 1 \text{ ms}$$

when the value of x is 2, the average waiting time of the processes is 1 millisecond.

- a7) how the base time quantum is related to the static priority:
find the base time quantum for the processes having highest, lowest and default static priority

(Ans) Base time quantum
(in milliseconds)

$$= \begin{cases} (140 - \text{static priority}) \times 20 & \text{if } SP < 120 \\ (140 - \text{static priority}) \times 5 & \text{if } SP \geq 120 \end{cases}$$

Base time quantum for highest priority

$$HP = 100$$

$$\begin{aligned} BTA &= (140 - (\text{static priority})) \times 20 \\ &= (140 - 100) \times 20 \\ &= 40 \times 20 = 800 \text{ ms} \end{aligned}$$

Base time quantum for lowest priority

$$LP = 139$$

$$\begin{aligned} BTA &= (140 - SP) \times 5 \\ &= (140 - 139) \times 5 = 5 \text{ ms} \end{aligned}$$

Base time quantum for default static priority

$$D.P. = 120$$

$$\begin{aligned} BTA &= (140 - 120) \times 5 \\ &= 20 \times 5 = 100 \text{ ms} \end{aligned}$$

(Q) what will be the dynamic priority of the following process?

<u>Process</u>	<u>static priority</u>	<u>Average sleep time</u>
P ₁	110	250 ms
P ₂	120	700 ms
P ₃	132	15

(Ans)	<u>Process</u>	<u>static priority</u>	<u>avg. sleep time</u>	<u>Bonus</u>
	P ₁	110	250 ms	2
	P ₂	120	700 ms	7
	P ₃	132	15	10

Dynamic priority,

$$P_j = \max(100, \min(\text{static priority} - \text{bonus} + 5, 139))$$

$$= \max(100, \min(110 - 2 + 5, 139))$$

$$= \max(100, \min(113, 139))$$

$$= \max(100, 113)$$

$P_1 = 113$

$$P_2 = \max(100, \min(\text{static priority} - \text{bonus} + 5, 139))$$

$$= \max(100, \min(120 - 7 + 5, 139))$$

$$= \max(100, \min(118, 139))$$

$$= \max(100, 118)$$

$P_2 = 118$

$$P_3 = \max(100, \min(132 - 10 + 5, 139))$$

$$= \max(100, \min(127, 139))$$

$$= \max(100, 127)$$

$P_3 = 127$

(a) what will be the minimum average sleep time for a process, so that the following processes will be considered as an interactive process by the scheduler in Linux system?

<u>Process</u>	<u>static priority</u>
P1	130
P2	108
P3	124
P4	132.

(Ans) A process is interactive if it satisfies the following condition.
 $\text{bonus} - 5 \geq \text{static priority} / 4 - 28$

P₁

$$\text{bonus} - 5 \geq \frac{130}{4} - 23$$

$$\Rightarrow \text{bonus} \geq \frac{130}{4} - 23 + 5$$

$$\Rightarrow \text{bonus} \geq 33 - 23$$

$$\Rightarrow \text{bonus} \geq 10$$

Avg sleep time of P₁ is 1 sec

P₂

$$\text{bonus} \geq \frac{108}{4} - 23$$

$$\Rightarrow \text{bonus} \geq 27 - 23$$

$$\Rightarrow \text{bonus} \geq 4$$

Average sleep time of P₂ is $\geq 400 < 500$ msec.

P₃

$$\text{bonus} \geq \frac{124}{4} - 23$$

$$\Rightarrow \text{bonus} \geq 31 - 23$$

$$\Rightarrow \text{bonus} \geq 8$$

Average sleep time of P₃ is $\geq 800 < 900$ msec

P₄

$$\text{bonus} \geq \frac{132}{4} - 23$$

$$\Rightarrow \text{bonus} \geq 33 - 23$$

$$\Rightarrow \text{bonus} \geq 10$$

so, avg. sleep time of P_1 is 1 sec.

Q10) which of the following process can not be considered as an interactive process scheduler in Linux system?

<u>Process</u>	<u>static priority</u>
P_1	136
P_2	104
P_3	116

(Ans) A process is interactive when the bonus is between 1 to 10.

P_1

$$\text{bonus} - 5 \geq \frac{\text{static priority}}{4} - 28$$

$$\Rightarrow \text{bonus} \geq \frac{136}{4} - 28 + 5$$

$$\Rightarrow \text{bonus} \geq \frac{136}{4} - 23 \geq 41$$

$\therefore P_1$ is not an interactive process.

P_2

$$\text{bonus} \geq \frac{104}{4} - 28 + 5$$

$$\Rightarrow \text{bonus} \geq 26 - 23$$

$$\Rightarrow \text{bonus} \geq 3$$

$\therefore P_2$ is an interactive process.

P₃

$$\text{bonus} \geq \frac{116}{4} - 28 + 5$$

$$\Rightarrow \text{bonus} \geq 29 - 23$$

$$\Rightarrow \text{bonus} \geq 6$$

$\therefore P_3$ is an interactive process.

Assignment #No. 3

On

Design of Operating Systems (CSE 4049)

Submitted by

Name : Saswat Mohanty

Reg. No. : 1941012407

Semester : 5th

Branch : CSE

Section : D

Session : 2021-2022

Admission Batch : 2019



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
FACULTY OF ENGINEERING & TECHNOLOGY (ITER)
SIKSHA 'O' ANUSANDHAN DEEMED TO BE UNIVERSITY
BHUBANESWAR, ODISHA – 751030**

P.S. Assignment - 3

(Q1) A counting semaphore s is initialized to 10. 8 wait operations followed by 6 signal operations are carried out. what is the final value of the semaphore s ?

(Ans) Initially, we have semaphore value = 10

after 8 wait operations, we get the semaphore value as $= 10 - 8 = 2$

now, we perform 6 v operation, here semaphore value will be $\underline{\underline{=}}$
 $= 2 + 6 = 8$

(Q2) Two concurrent processes P and Q are accessing their critical sections by using Boolean variables S and T as follows:

Process P	Process Q
$\text{while}(\text{true})$ { // Entry section? $\text{Point}(1);$ $\text{Point}(1);$ // Exit section? }	$\text{while}(\text{true})$ { // Entry section? $\text{Point}(0);$ $\text{Point}(0);$ // Exit section? }

compare the entry section and exit section of Process P and Q with suitable semaphore operations using the two Boolean semaphores S and T. Also suggest the initial values of S and T such that execution of the processes will print the sequence 00110011...

(Ans) initial value $S = 0, T = 1$

Process P	Process Q
<pre> while (true) { wait(s); print(1); print(1); signal(t); } </pre>	<pre> while (true) { wait(t); print(0); print(0); signal(s); } </pre>

- Q3) Two concurrent processes P and Q are executing the following instructions, synchronizing the execution of P and Q with suitable semaphore operations and choose initialization of the semaphore values, so that the final outcome will be in the following order:-
- a) 1 3 2 4 b) 3 1 2 4

Process P	Process Q
<pre> print(1); print(2); </pre>	<pre> print(3); print(4); </pre>

(Ans) a), $s = 1, t = 0$

process P	process Q
<pre> wait(s); print(1); signal(t); wait(s); print(2); signal(t); </pre>	<pre> wait(t); print(3); signal(s); wait(t); print(4); signal(s); </pre>

b) $S = 0, T = 1$

Process P	Process Q
wait (S);	wait (T);
print (3);	print (3);
print (2);	signal (S);
signal (T);	wait (T);
	print (4);
	signal (S);

Q4) Assume the following 3 concurrent processes that use 3 binary semaphores s_0, s_1, s_2 initialized as $s_0 = 1, s_1 = 0, s_2 = 0$. How many maximum and minimum number of times will process P0 print '0'? Justify your answer

Process P0	Process P1	Process P2
<pre>while (true) { wait (s0); print (0); signal (s1); signal (s2); }</pre>	<pre>wait (s1); signal (s0);</pre>	<pre>wait (s2); signal (s0);</pre>

(Ans) Minimum = 2

Maximum = 3

Q5) Let 4 concurrent processes P_1, P_2, P_3, P_4 are accessing their critical sections by using Boolean semaphores s_1, s_2, s_3 and s_4 . Write the entry section and exit section of all processes using the semaphores with suitable initialization, such that P_1 will complete its critical section before P_2 and P_3 , P_2 and P_3 will complete their critical section in any order before P_4 .

(Ans) $s_1 = 0$, $s_2 = 0$, $s_3 = 0$, $s_4 = 0$

P_1 wait(s_1); <input checked="" type="checkbox"/>	P_2 wait(s_2); <input checked="" type="checkbox"/>	P_3 wait(s_3); <input checked="" type="checkbox"/>	P_4 wait(s_4); <input checked="" type="checkbox"/>
signal(s_2);	signal(s_3);	signal(s_4);	signal(s_1);

oR

P_1 wait(s_1); <input checked="" type="checkbox"/>	P_2 wait(s_2); <input checked="" type="checkbox"/>	P_3 wait(s_3); <input checked="" type="checkbox"/>	P_4 wait(s_4); <input checked="" type="checkbox"/>
signal(s_3);	signal(s_4);	signal(s_2);	signal(s_1);

Q6) Assume that val is an atomic integer in a Linux system. What is the value of val after the following operations have been completed?

```
atomic_set(&val, 10);
atomic_sub(8, &val);
atomic_inc(&val);
atomic_inc(&val);
atomic_add(6, &val);
atomic_sub(3, &val);
```

(Ans) As given in the question we will find the value of val,

```
atomic_set(&val, 10);
→ 10
atomic_sub(8, &val);
→ 10 - 8 = 2

atomic_inc(&val);
→ 2 + 1 = 3

atomic_inc(&val);
→ 3 + 1 = 4
```

atomic - add (6, & val);

$$\rightarrow 6 + 4 = 10$$

atomic - sub (3, & val);

$$\rightarrow 10 - 3 = 7$$

a) Define the compare-and-swap hardware instruction. Specify a solution to critical section problem using compare-and-swap instruction and explain how the solution will satisfy all the three requirements.

(Ans) boolean compareAndSwap (boolean *target, boolean expected, boolean new)

{

boolean nw = *target;

If (*target == expected)

*target = new;

return nw;

}

A solution to critical section problem using compare-and-swap instruction which satisfies all three requirements.

do

{

waiting [\hat{i}] = T

key = T

while (key == T && waiting [\hat{i}] == T)

key = (lock, F, T);

waiting [\hat{i}] = F;

$\hat{j} = (\hat{i} + 1) \% n$;

while ($\hat{j} \neq \hat{i}$ && waiting [\hat{j}] == F).

$\hat{j} = (\hat{j} + 1) \% n$;

if ($\hat{i} == \hat{j}$)

lock = False;

```

    else
        waiting [if] = False.
    [RS]
} while (TRUE);

```

88) write a monitor solution for the bounded buffer producer consumer problem.

(Ans) Monitor PC

```

{
    int c;
    condition full, empty;
    void put-item (item p)
    {
        if (c == N)
            full.wait();
        insert (item p);
        c = c + 1;
        if (c == 1)
            empty.signal();
    }
    void get-item()
    {
        if (c == 0)
            empty.wait();
        Remove (item p);           // item p = Buffer [c];
        c = c - 1;                out = (out + 1)/N
        if (c == N - 1)
            full.signal();
    }
    initialization code()
    {
        int c = 0;
    }
}

```

producer()

{

 while(1)

{

 producer(item p);

 PC.put-item(item p);

}

}

consumer()

{

 while(1)

{

 PC.get-item();

 consumer(item p);

}

}

- Q) write a solution using semaphores for a Reader's Writer's problem in which writer has higher priority than reader. once a writer is ready, that writer performs its write as soon as possible. in other words, if a writer is waiting to access the object, no new reader may start reading.

(Ans) semaphore mutext = 1

int RW = 0

int AR = 0

int WR = 0

int WW = 0

int OKread = 0

int OKwrite = 0

Reader process	Writer process
reader() { while(1) ? }	writer() { while(1) ? }

```

    wait (Mutex);
    if (AR + WW == 0)
    {
        signal (OKread);
        AR = AR + 1;
    }
    else
        WR = WR + 1;
    signal (Mutex);
    wait (OK read);
    //Reading the database
    //Exit code
    wait (Mutex);
    AR = AR - 1;
    if (AR == 0 & & WW > 0)
    {
        signal (OKwrite);
        AW = AW + 1;
        WW = WW - 1;
    }
    signal (Mutex);
}
}

    wait (Mutex);
    if (AR + AW == 0)
    {
        signal (OKwrite);
        AW = AW + 1;
    }
    else
        WW = WW + 1;
    signal (Mutex);
    wait (OK write);
    //write the database
    wait (Mutex);
    AW = AW - 1;
    if (WW > 0)
    {
        signal (OKwrite);
        AW = AW + 1;
        WW = WW - 1;
    }
    else
        while (WR > 0)
    {
        signal (OKread);
        AR = AR + 1;
        WR = WR - 1;
    }
    signal (Mutex);
}
}

```

- Q10) The sleeping - Barber Problem. A barbershop consists of a waiting room with n chairs and a barbershop with one barber chair. If there are no customers to be served, the barber goes to

sleep. If a customer enters the barbershop and all chairs are occupied,⁹ then the customer leaves the shop. If the barber is busy but chairs are available, then the customer sits in one of the free chairs. If the barber is asleep, the customer wakes up the barber.

- a) write a solution using semaphores to coordinate the barber & the customers.
- b) write a solution using monitor to coordinate the barber & the customer.

(Ans) Barber = 0

$$FC = N$$

$$cus = 0$$

$$M = 1$$

Barber code

```
while(1)
{
    wait(cus);
    wait(M);
    FC++;
    signal(Barber);
    signal(M);
}
```

Customer code

```
while(1)
{
    wait(M);
    if(FC > 0)
    {
        FC--;
        signal(cus);
        signal(M);
        wait(Barber);
    }
    else
    {
        signal(M);
    }
}
```

Monitor solution :-

Hairdresshop

waiting = 0;

customer = 0;

barber = 0

procedure seek - customer()

begin if waiting = 0

wait (customers);

waiting = waiting - 1;

signal (barber);

end seek - customer;

procedure get - haircut()

begin if waiting < 0 then

{

waiting = waiting + 1;

signal (customers);

wait (barber);

}

end get - haircut;

end Hairdresshop;

Assignment #No. 4

On

Design of Operating Systems (CSE 4049)

Submitted by

Name : Saswat Mohanty

Reg. No. : 1941012407

Semester : 5th

Branch : CSE

Section : D

Session : 2021-2022

Admission Batch : 2019



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
FACULTY OF ENGINEERING & TECHNOLOGY (ITER)
SIKSHA 'O' ANUSANDHAN DEEMED TO BE UNIVERSITY
BHUBANESWAR, ODISHA – 751030**

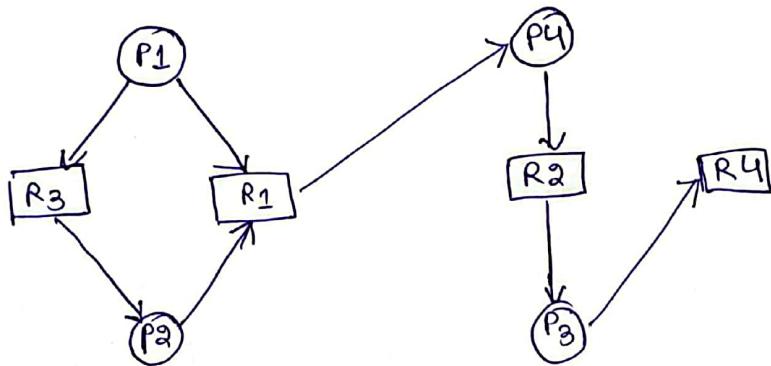
P.S. Assignment - 4

Q1) consider a system with four processes P_1, P_2, P_3 and P_4 , resources types R_1, R_2, R_3, R_4 each one with a single instance. Draw the resource allocation graph corresponding to following resource allocation state:

- P_1 requests for R_3 and R_1
- P_2 holds R_3 and requests for R_1
- P_3 holds R_2 and requests for R_1
- P_4 holds R_1 and requests for R_2

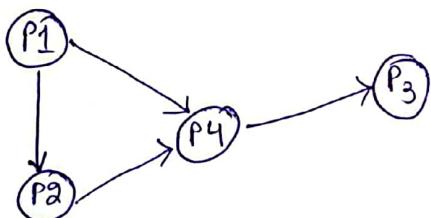
Using wait graph, check whether the system is in deadlock or not? If so how many processes are involved in deadlock?

(Ans)



The system is not in deadlock. [No cycle]

wait for graph



Q2) consider a system with 12 tape drives & 3 processes: P_0, P_1, P_2 . P_0 requires 4 tape drives, P_1 requires 10 tape drives, P_2 requires 9 tape drives for completing their task. suppose at time t_0 , P_0 is holding 2 tape drives, P_1 is holding 5 tape drives and P_2 is holding 2 tape drives. Then check whether the current resource allocation state is safe or not. If yes specify the safe sequence.

(Ans)

Given :- 12 tape drives \rightarrow Resource

Process \rightarrow P0, P1, P2

Request

P0	4
P1	10
P2	9

	<u>Allocated</u>	<u>Req.</u>	<u>Available</u>	<u>req < available</u>	<u>New available</u>
P0	2	2	3	True	5
P1	5	5	·	True	10
P2	2	7	·	True	12

Yes, current allocation is safe

Safe sequence :- P0, P1, P2

- Q3) consider a system has 10 units of a resource type and 5 resources
The current resource allocation state is given as follows:

Process	Used	Max
P0	2	5
P1	1	6
P2	2	6
P3	1	2
P4	1	4

If P2 will request for 2 more instances of the same resource type,
check the request can be granted immediately or not?

(Ans)

③

<u>Process</u>	<u>Used</u>	<u>Max</u>	<u>Req</u>	<u>Available</u>	<u>Req < available</u>	<u>New available</u>
P0	2	5	3	3	True	5
P1	1	6	5		True	6
P2	2	6	4		True	8
P3	1	2	1		True	9
P4	1	4	3		True	10

safe sequence :- P0, P1, P2, P3, P4

P2 will be granted with 2 more additional instances if it request as there are 3 more instance available.

Q4) consider a system with two resources A and B. A has 6 instances and B has 3 instances. Can the system execute the following processes without any deadlock? If yes specify the safe sequence.

<u>Process</u>	<u>Allocation</u>		<u>Max</u>	
	<u>A</u>	<u>B</u>	<u>A</u>	<u>B</u>
P0	1	1	2	2
P1	1	0	4	2
P2	1	0	3	2
P3	0	1	1	1
P4	2	1	6	3

(Ans)

<u>Process</u>	<u>Allocation</u>	<u>Max</u>	<u>Need</u>	<u>available</u>	<u>need < available</u>	<u>New available</u>
	<u>A</u> <u>B</u>	<u>A</u> <u>B</u>				
P0	1 1	2 2	1 1	1 0	False	True
P1	1 0	4 2	3 2		F T	4 2
P2	1 0	3 2	2 2		F T	3 2
P3	0 1	1 1	1 0		T T	1 1
P4	2 1	6 3	4 2		F F	6 3

(5)

Safe sequence :- $\langle P_3, P_0, P_2, P_1, P_4 \rangle$

Q5) Consider the following resource allocation state with 3 processes & 3 resources :

	Allocation			Max		
	X	Y	Z	X	Y	Z
P0	0	0	1	7	4	3
P1	3	2	0	6	2	0
P2	2	1	1	3	3	3

there are 3 instances of type X, 2 instances of type Y and 2 instances of type Z, still available.

a) find the content of the need matrix.

b) Is the system in a safe state?

c) If P0 will request for 2 more instances of type Z, can the request be granted immediately or not?

(Ans) a)

	Allocation			Max			Need			Available			New Available		
	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
P0	0	0	1	7	4	3	7	4	2	3	2	2	8	5	4
P1	3	2	0	6	2	0	3	0	0				6	4	2
P2	2	1	1	3	3	3	1	2	2				8	5	3

b) safe sequence :- $\{P_1, P_2, P_0\}$

c) No, P0 can't be assigned with 2 more instance of Z as P0 already needs 2 instance of Z and are available with only 3 instances of Z

Q6) consider the following resource allocation state with 4 processes & 4 resources, and an available vector of $\langle 0100 \rangle$

	Allocation				Request			
	A	B	C	D	A	B	C	D
P0	0	1	1	0	0	0	1	0
P1	0	1	0	1	1	0	0	1
P2	1	2	0	0	0	0	0	1
P3	0	0	1	2	0	0	0	0
P4	1	0	1	0	0	1	0	0

- a) find the initial number of instances available for each resource type
- b) check whether the system is deadlock free or not
- c) If P0 is assigned with 1 more instance of type B, then check whether the system is deadlock free or not.

(Ans)

	<u>Allocation</u>				<u>Request</u>				<u>New available</u>			
	A	B	C	D	A	B	C	D	A	B	C	D
P0	0	1	1	0	0	0	1	0	1	2	3	2
P1	0	1	0	1	1	0	0	1	1	2	3	3
P2	1	2	0	0	0	0	0	1	2	5	3	3
P3	0	0	1	2	0	0	0	0	0	0	1	2
P4	1	0	1	0	0	1	0	0	1	1	2	2

a) available

$$A \rightarrow 0$$

$$B \rightarrow 1$$

$$C \rightarrow 0$$

$$D \rightarrow 0$$

b) safe sequence :- $\langle P_3, P_4, P_0, P_1, P_2 \rangle$

deadlock free.

c) $P_0 \rightarrow$ 1 more type B

	<u>Allocation</u>				<u>Request</u>				<u>available</u>				<u>new available</u>			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	2	1	0	0	0	1	0	0	0	0	0	0	2	2	2
P_1	0	1	0	1	1	0	0	1						2	5	3
P_2	1	2	0	0	0	0	0	1						1	4	2
P_3	0	0	1	2	0	0	0	0						2	4	3
P_4	1	0	1	0	0	1	0	0						2	4	3

safe sequence :- $\langle P_3, P_0, P_2, P_4, P_1 \rangle$

deadlock free

Assignment #No. 5

On

Design of Operating Systems (CSE 4049)

Submitted by

Name : Saswat Mohanty

Reg. No. : 1941012407

Semester : 5th

Branch : CSE

Section : D

Session : 2021-2022

Admission Batch : 2019



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
FACULTY OF ENGINEERING & TECHNOLOGY (ITER)
SIKSHA 'O' ANUSANDHAN DEEMED TO BE UNIVERSITY
BHUBANESWAR, ODISHA – 751030**

(1)

P.S. Assignment - 5

Q1) Given four memory locations of 200K, 600K, 400K, 700K (in order). How the First-fit, Best-fit, Worst-fit algorithms place processes of 312K, 517K, 212K, 526K (in order). Which algorithm makes the most efficient use of memory?

(Ans) Memory partition \rightarrow 200K, 600K, 400K, 700K

Processes \rightarrow 312K, 517K, 212K, 526K

First fit

312K in 600K

517K in 700K

212K must wait

526K must wait

Best fit

312K in 400K

517K in 600K

212K in 700K

526K must wait

Worst fit

312K in 700K

517K in 600K

212K in 400K

526K must wait.

Q2) Using Page size of 16 bytes a physical memory of 2048 bytes and logical memory of 128 bytes.

- Find the number of bits required to represent logical address
- Find the number of bits required to represent logical addresses
- Find the number of entries in the page table.

(2) a) find the total no. of frames

c) find the physical address of the logical address 80 with the following page table :

8
6
5
2
3
1
4
7

(Ans) a) No. of bits required to represent logical address

$$2^n = 128 \text{ bytes}$$

$$\Rightarrow n = 7$$

b) No. of bits required to represent physical address

$$2^n = 2048 \text{ bytes}$$

$$\Rightarrow n = 11$$

c) No. of entries in page table

$$2^{17} / 16 \text{ bytes}$$

$$\Rightarrow 2^{17} / 2^{14} = 2^3$$

d) Total no. of frames = $2^{21} / 2^{14} = 2^7$

e)

0	3
1	6
2	5
3	2
4	3
5	1
6	4
7	7

$$20 / 16 = 1$$

$$20 \% 16 = 4$$

$$6 \times 16 + 4 = 100$$

(3) Q3) How many numbers of pages are required for a process having size 8005 bytes with a page size of 200 bytes?

(Ans) No. of pages = $\frac{\text{Process size}}{\text{Page size}}$

$$= \frac{8005}{200} = 40 \text{ page} + 5 \text{ bytes}$$

Q4) With a page size of 2048 bytes, find the amount internal fragmentation arises for storing a process of size 72766 bytes.

(Ans) No. of page = 35 page + 1086 bytes

$$\text{Internal fragmentation} = 2048 - 1086 = 962 \text{ bytes}$$

Q5) Consider a machine with 64mb physical memory and a 32-bit virtual address space. If the page size is 4KB, how many entries will there be in a conventional single level page table and in an inverted page table?

(Ans) No. of pages = $2^{32}/2^{12} = 2^{20}$

$$\text{No. of entries} = 1024 \text{ conventional single-level}$$

$$2^{26} = \text{Total physical memory}$$

$$2^{12} \text{ page size} = \text{frame size}$$

$$2^{29}/2^{12} = 2^{17} = \text{Total no. of frame}$$

$$\text{Number of entries} = 128k \text{ inverted page table}$$

Q6) In paging scheme, if the page size is 2KB and process size is 83412 bytes. Then find the number of pages required and the size of internal fragmentation.

(Ans) No. of page = $\frac{83412 \text{ bytes}}{2000 \text{ bytes}} = 41 \text{ page} + 1412 \text{ bytes}$

$$\text{Internal fragmentation} = 2000 - 1412 = 588 \text{ bytes}$$

07) A specific editor has 200K of program text, 15K of initial stack, 50K of initialized data, and 70K of bootstrap code. If five processes are started simultaneously, how much physical memory is needed if shared program text is used? (4)

(Ans) Total physical memory needed = $200 + 15 + 50 + 70 \approx 335K$

08) If the hit ratio of a translational look A-side Buffer (TLA) is 80% & it takes 15 nanoseconds to search the TLB and 150 nanoseconds to access the main memory, what is the effective access time?

(Ans) $d = 80\%$, $E = 15 \text{ ns}$

150 ns for memory access

$$EAT = 0.80(15 + 150) + 0.20(15 + 100 + 100) = 175 \text{ ns}$$

09) A computer system implements 8KB pages and a 32-bit physical address space. Each page table entry contains a valid bit, a dirty bit, three permissions bits, and the frame numbers. If the maximum size of the page table of a process is 24 kilobytes. Find the length of the virtual address supported by the system in bits.

(Ans) Page size = 2^{13} bytes = framesize

$$\text{No. of frame} = 2^{32} / 2^{13} = 2^{19}$$

All entries can be addressed by 23 bits

$$8 \times 2^{20} = 2^{23}$$

$$\text{Virtual address} = 23 + 13 = 36 \text{ bits}$$

10) Consider the byte addressable system with physical address space of 128 byte, logical address space of 64 bytes & a page size of 8 byte. The page-table is specified as follows.

4
5
1
3

- a) Find the number of bits required to represent logical address.
- b) Find the number of bits required to represent the physical address.
- c) Find the physical address of the logical address 12
- d) Find the physical address in hexadecimal representation of the logical address $(35)_X$

(Ans) a) No. of bits required to represent logical address

$$2^n = 64 \text{ bytes}$$

$$\Rightarrow n = 6$$

b) $2^n = 128 \text{ bytes}$

$$\Rightarrow n = 7$$

c)

0	4
1	5
2	1
3	3

$$12 / 16 = 0$$

$$12 \% 16 = 12$$

$$\therefore 16 \times 12 + 12 = 204$$

d) $35 \rightarrow 0011 \quad 0101$

Page size $16 \text{ KB} = 2^{14}$. So 14 bits are offset.

$$0100 \quad 0000 \quad 0011 \quad 0101 \rightarrow 4035$$

e) consider the following segment table

segment	Base	Length
0	219	600
1	2300	100
2	-90	110
3	1327	400
4	1950	50

what are the physical addresses for the following logical address? (6)

a) 0, 430

b) 1, 10

c) 2, 100

d) 2, 500

(Ans) a) 0, 430

$$219 + 430 = 649$$

b) 1, 10

$$2300 + 10 = 2310$$

c) 2, 100

$$90 + 100 = 190$$

d) 2, 500

Illegal reference