List of Experiments

1. Study of basic Commands in Linux Operating System.
2. Write a program to implementation of Thread and Process.
3. Write a C program to implementation of producer-consumer problem using Semaphores.
4. Write a C program to simulate the concept of Dining-philosophers problem.
5. Write a C program to simulate the following non-preemptive CPU scheduling algorithms to find turnaround time and waiting time.
   a) FCFS b) SJF c) Priority
6. Write a C program to simulate the following preemptive CPU scheduling algorithms to find turnaround time and waiting time.
   a) Round Robin b) Priority
7. Program to Simulate Bankers Algorithm for Dead Lock Avoidance.
8. Program to Simulate Bankers Algorithm for Dead Lock Prevention.
9. Program to Simulate the MVT and MFT memory management techniques.
10. Simulate paging technique of memory management.
11. Write a C program to simulate the FIRST-FIT contiguous memory allocation technique.
12. Write a C program to simulate the BEST FIT contiguous memory allocation technique.
13. Write a C program to simulate the WORST-FIT contiguous memory allocation technique.
14. Write a C program to simulate FIFO page replacement algorithm.
15. Write a C program to simulate LRU page replacement algorithm.
16. Write a C program to simulate LFU page replacement algorithm.
17. Write a C program to simulate Optimal page replacement algorithm.
18. Write a C program to simulate the following file organization techniques
    a) Single level directory b) Two level directory c) Hierarchical
19. Write a program to Simulate all file allocation strategies
    a) Sequential b) Indexed c) Linked
20. Write a C program to simulate disk scheduling algorithms.
    a) FCFS b) SCAN c) LOOK

Note: Generate table of contents/Index page and Use Software C/C++….

Samples

EXPERIMENT NO. 1

UNIX COMMANDS

AIM / TITLE: To study and execute the commands in unix.

COMMAND:

1.Date Command:

This command is used to display the current data and time.

Syntax:

```
$date
$date +%ch
```

Options: -

a = Abbrevated weekday.

A = Full weekday.

b = Abbrevated month.

B = Full month.

c = Current day and time.

C = Display the century as a decimal number.

d = Day of the month.

D = Day in „mm/dd/yy‟ format

h = Abbrevated month day.

H = Display the hour.

L = Day of the year.

m = Month of the year.

M = Minute.

P = Display AM or PM

S = Seconds

T = HH:MM:SS format

u = Week of the year.

y = Display the year in 2 digit.

Y = Display the full year.

Z = Time zone .

To change the format:

Syntax:

$date „+%H-%M-%S‟

2.Calender Command:

This command is used to display the calendar of the year or the particular month of calendar year.

Syntax:

a.$cal <year>

b.$cal <month> <year>

Here the first syntax gives the entire calendar for given year & the second Syntax gives the calendar of reserved month of that year.

3.Echo Command:

This command is used to print the arguments on the screen.

Syntax: $echo <text>

Multi line echo command:

To have the output in the same line, the following commands can be used.

Syntax: $echo <text\>text

To have the output in different line, the following command can be used.

Syntax: $echo "text

>line2

>line3"

4.Banner Command:

It is used to display the arguments in „#‟ symbol.

Syntax: **$banner <arguments>**

……………………..

EXPERIMENT NO. 2

CPU SCHEDULING ALGORITHMS

A). FIRST COME FIRST SERVE:

Objective: To write a c program to simulate the CPU scheduling algorithm First Come First Serve (FCFS)

Theory/ Algorithm:

Step 1: Start the process

Step 2: Accept the number of processes in the ready Queue

Step 3: For each process in the ready Q, assign the process name and the burst time Step

Step 4: Set the waiting of the first process as 0‘and its burst time as its turnaround time Step

Step 5: for each process in the Ready Q calculate

a). Waiting time (n) = waiting time (n-1) + Burst time (n-1)

b). Turnaround time (n)= waiting time(n)+Burst time(n)

Step 6: Calculate

a) Average waiting time = Total waiting Time / Number of process

b) Average Turnaround time=Total Turnaround Time /Number of process

Step 7: Stop the process

Source code:

```
#include<stdio.h>
#include<conio.h>
main()
{
int bt[20], wt[20], tat[20], i, n;
float wtavg, tatavg;
clrscr();
printf("\n Enter the number of processes-- ");
scanf("%d", &n);
for(i=0;i<n;i++)
{
printf("\n Enter Burst Time for Process %d --", i);
scanf("%d", &bt[i]);
}
wt[0] = wtavg = 0;
tat[0] = tatavg = bt[0];
for(i=1;i<n;i++)
{
wt[i] = wt[i-1] +bt[i-1];
tat[i] = tat[i-1] +bt[i];
wtavg = wtavg + wt[i];
tatavg = tatavg + tat[i];
}
printf("\t PROCESS \t BURST TIME \t WAITING TIME \t TURNAROUND TIME \n");
for(i=0;i<n;i++)
printf("\n \t P%d \t\t %d \t \t %d \t \t %d", i, bt[i], wt[i], tat[i]);
printf("\n Average Waiting Time--%f", wtavg/n);
printf("\n Average Turnaround Time -- %f", tatavg/n);
getch();
}
```

Input

```
Enter the number of processes --      3
Enter Burst Time for Process 0--      24
Enter Burst Time for Process 1 --     3
Enter Burst Time for Process 2 --     3
```

Output

```
PROCESS     BURST TIME        WAITING TIME     TURNAROUNDTIME
P0          24                0                        24
P1          3                 24                       27
P2          3                 27                       30
```

```
Average Waiting Time--17.000000
Average Turnaround Time--27.000000
```