**SCHOOL OF COMPUTING AND IT**

**II B.Tech. IV Semester; First- Sessional Examination, Feb-2018**

**Branch: CSE / IT /CCE**

**Subject Code: CS1401**

**Subject Name : Operating Systems**

**Max. Marks: 15 Duration: 1 hour**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Instructions:**

* All questions are compulsory
* Closed Book Exam
* Missing data if any can be suitably assumed

1. What is the main advantage of microkernel approach to system design? How do user programs and system services interact in microkernel architecture? What are the disadvantages of using microkernel approach? [3]

**Ans:** Benefits typically include the following (a) adding a new service does not require modifying the kernel,

(b) it is more secure as more operations are done in user mode than in kernel mode,

(c) a simpler kernel design and functionality typically results in a more reliable and easily portable kernel. [1]

Interaction: by message passing (a type of IPC mechanism) [1]

Disadvantage: performance reduced due to frequent switches between user mode and kernel mode. The primary disadvantage of the microkernel architecture are the overheads associated with inter-process communication and the frequent use of the operating system’s messaging functions in order to enable the user process and the system service to interact with each other. [1]

1. Write a program in ‘C’ (Linux OS) in which a parent forks a child and child forks a grandchild. The program should display the output as **“My process ID is A, my parent’s ID is B and my grandfather’s ID is C”**, where A, B and C are the process ID’s. [3]

Ans: Program is given below. If the student has created the correct process tree, then 1.5 marks may be given.

#include <stdio.h>

main()

{

int p1,p2;

p1=fork();

if(p1>0)

{

wait();

printf("My grand parents ID is %d\n",getpid());

}

else

{

p2=fork();

if(p2==0)

{

printf("My Id is %d and ",getpid());

}

else

{

wait();

printf(" and my parents Id is %d and ",getpid());}

}

}

}

1. Write a program in ‘C’ (Linux OS) in which the parent process sends the length and breadth of a rectangle and child process computes the area of rectangle. The communication between the parent and the child is through pipe. [3]

Program is given below. If the student has correctly done pipe creation, after that process creation and does the task of sending and receiving data then 1.5 marks may be given.

main()

{

int fd[2], n1,n2, area, length, breadth;

char len[10],brth [10];

pipe(fd);

if(fork()!=0)

{

close(fd[0]);

n1=read (0, len, 10);

write (fd[1], len, n1);

n2 = read (0, brth, 10);

write (fd[1], brth, n2);

close(fd[1]);

}

else

{

close(fd[1]);

read (fd[0], len, 10);

read (fd[0], brth, 10);

length = atoi (len);

breadth = atoi (brth);

area = length \* breadth;

printf("area is %d \n ", area);

close (fd[0]);

}

}

1. Consider the following set of processes, with the length of the CPU-burst time given in milliseconds:

|  |  |  |
| --- | --- | --- |
| Process | Arrival Time | CPU Burst Time |
| P1 | 0 | 2 |
| P2 | 1 | 6 |
| P3 | 4 | 1 |
| P4 | 7 | 4 |
| P5 | 8 | 3 |

1. Draw three Gantt charts illustrating the execution of these processes using First Come First Serve (FCFS), Shortest Remaining Time First (SRTF) and Round Robin (RR) with time quantum = 1, scheduling. Assume that context switch overhead is 0.
2. What is the turnaround time of each processes for SRTF and RR scheduling algorithms?
3. What is the response time of each processes for SRTF and RR scheduling algorithms?
4. How many context switches are required in SRTF Scheduling? [1.5 +2 +2+0.5 = 6]

**Ans:**

1. **FCFS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0-P1 | 2-P2 | 8-P3 | 9-P4 | 13-P5-16 |

**SRTF**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0-P1 | 2-P2 | 4-P3 | 5-P2 | 9-P5 | 12-P4-16 |

**RR**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0-P1 | 1-P2 | 2-P1 | 3-P2 | 4-P3 | 5-P2 | 6-P2 | 7-P4 | 8-P2 | 9-P5 | 10-P4 | 11-P2 | 12-P5 | 13-P4 | 14-P5 | 15-P4-16 |

1. **Turn-around time**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Scheduler | Process 1 | Process 2 | Process 3 | Process 4 | Process 5 | Average turn-around time |
| SRTF | 2 | 8 | 1 | 9 | 4 | 4.8 |
| RR | 3 | 11 | 1 | 9 | 7 | 6.2 |

1. **Waiting Time**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Scheduler | Process 1 | Process 2 | Process 3 | Process 4 | Process 5 |
| SRTF | 0 | 2 | 0 | 5 | 1 |
| RR | 1 | 5 | 0 | 5 | 4 |

1. **No. of context switches in SRTF 5**