RAMAPURAM SRM

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

RAMAPURAM CAMPUS

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CONTINUOUS LEARNING ASSESSMENT-1

Sub Code/Name:18CSC205J – Operating Systems **Set : ODD**

Class : II Year / IV Sem / B.Tech (CSE, AIML,IOT,BDA,CS)

Date :18 .04.2022

Max Marks :25

Answer Key

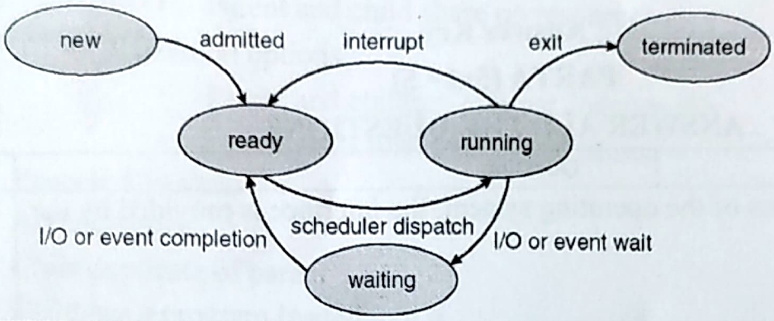
PARTA (5x1= 5)

ANSWER ALL THE QUESTIONS

Q.No.	Question	Marks
1	To access the services of the operating system, the interface is provided by the ___________ a) Library **b) System calls** c) Assembly instructions d) API	1
2	If a process fails, most operating system write the error information to a ______ a) new file b) another running process **c) log file** d) history file	1
3	In a timeshare operating system, when the time slot assigned to a process is completed, the process switches from the current state to? a) Suspended state b) Terminated state **c) Ready state** d) Blocked state	1
4	The portion of the process scheduler in an operating system that dispatches processes is concerned with ____________ a) assigning ready processes to waiting queue b) assigning running processes to blocked queue **c) assigning ready processes to CPU** d) assigning running processes to waiting queue	1
5	In Unix, which system call creates the new process? a) create **b) fork** c) new d) pipe	1

PARTB (2x4= 8)

ANSWER ANY TWO

Q.No.	Question	Marks
6.	Define Process. What is the various process state? A process is basically a program in execution. The execution of a process must progress in a sequential fashion. A process is defined as an entity which represents the basic unit of work to be implemented in the system. Various process states : New, Ready, Running, Blocked, Terminated  new admitted interrupt exit terminated ready running scheduler dispatch I/O or event completion I/O or event wait waiting	4
7.	Describe the actions taken by a kernel to context switch between processes. The steps involved in context switching are as follows – • Save the context of the process that is currently running on the CPU. Update the process control block and other important fields. • Move the process control block of the above process into the relevant queue such as the ready queue, I/O queue etc. • Select a new process for execution. • Update the process control block of the selected process. This includes updating the process state to running. • Update the memory management data structures as required. • Restore the context of the process that was previously running when it is loaded again on the processor. This is done by loading the previous values of the process control block and registers.	4
8.	Explain Critical Section Problem. Give an example. ■ When a process executes code that manipulates shared data (or resources), we say that the process is in its critical section (CS) for that shared data ■ We must enforce mutual exclusion on the execution of critical sections. ■ Only one process at a time can be in its CS (for that shared data or resource). Each process must ask permission to enter critical section in **entry section,** may follow critical section with **exit section**, then **remainder section** Example : Producer Consumer problem	4

PART C (1x12= 12)

ANSWER ANY ONE

Q.No.	Question	Marks

A) Explain the Evolution of Operating Systems.

A major OS will evolve over time for a number of reasons:

- Hardware upgrades
- New types of hardware
- New services
- Fixes

Serial Processing

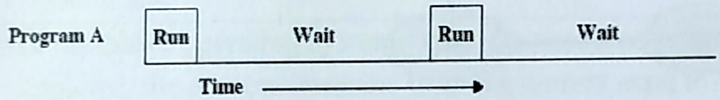
- No operating system. Programmers interacted directly with the computer hardware
- Computers ran from a console with display lights, toggle switches, some form of input device, and a printer
- Users have access to the computer in "series"

Simple Batch Systems

- Early computers were very expensive
  - Important to maximize processor utilization
- Monitor
  - User no longer has direct access to processor
  - Job is submitted to computer operator who batches them together and places them on an input device
  - Program branches back to the monitor when finished

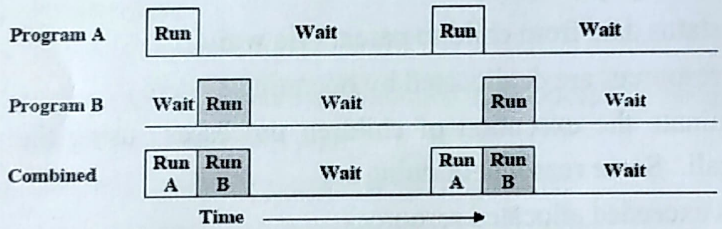
Multi programmed Batch Systems

9. **Uniprogramming** 12



(a) Uniprogramming

Multiprogramming



(b) Multiprogramming with two programs

Time-Sharing Systems

- Can be used to handle multiple interactive jobs
- Processor time is shared among multiple users
- Multiple users simultaneously access the system through terminals, with the OS interleaving the execution of each user program in a short burst or quantum of computation

(or)

B) Explain the Operations Process creation and Process termination. Also specify the system calls used in Process creation and termination.

Operations on Processes

- **Parent** process create **children** processes, which, in turn create other processes, forming a **tree** of processes
- Generally, process identified and managed via a **process identifier (pid)**
- Resource sharing options
  - Parent and children share all resources
  - Children share subset of parent's resources
  - Parent and child share no resources
- Execution options
  - Parent and children execute concurrently
  - Parent waits until children terminate

Process Creation

Address space

Child duplicate of parent

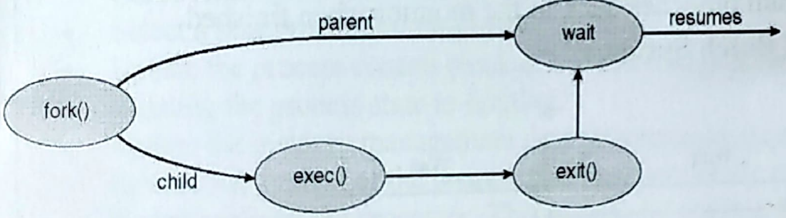
Child has a program loaded into it

UNIX examples

fork() system call creates new process

12

exec() system call used after a **fork()** to replace the process' memory space with a new program



Process Termination

- Process executes last statement and then asks the operating system to delete it using the **exit()** system call.
  - Returns status data from child to parent (via **wait()**)
  - Process' resources are deallocated by operating system
- Parent may terminate the execution of children processes using the **abort()** system call. Some reasons for doing so:
  - Child has exceeded allocated resources
  - Task assigned to child is no longer required
  - The parent is exiting and the operating systems does not allow a child to continue if its parent terminates

HOD/CSE

Prepared By :
N. Sug 19/4/22