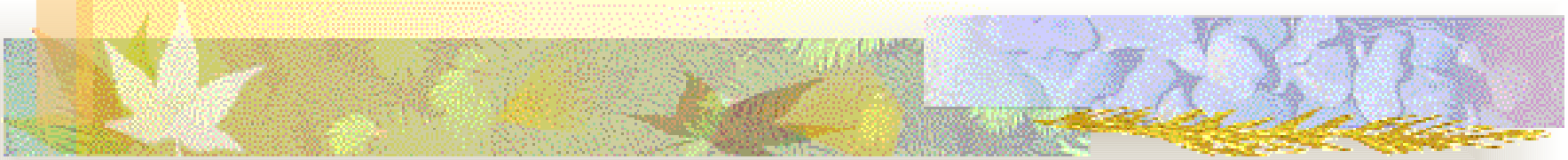
Guidelines for Internal-I



CSEN3103-Sec A, Prepared by
Nilina Bera, CSE, HIT

- The bounded buffer problem is also known as :

a) Readers – Writers problem
b) Dining – Philosophers problem
c) Producer – Consumer problem
d) None of these

Answer: c

- In the bounded buffer problem, there are the empty and full semaphores that :

a) count the number of empty and full buffers
b) count the number of empty and full memory spaces
c) count the number of empty and full queues
d) None of these

Answer: a

In the bounded buffer problem :

a) there is only one buffer

b) there are n buffers (n being greater than one but finite)

c) there are infinite buffers

d) the buffer size is bounded **Answer: b**

To ensure difficulties do not arise in the readers – writers problem, _____ are given exclusive access to the shared object.

a) readers

b) writers **Answer: b**

c) None of these

The dining – philosophers problem will occur in case of :

a) 5 philosophers and 5 chopsticks

b) 4 philosophers and 5 chopsticks

c) 3 philosophers and 5 chopsticks

d) 6 philosophers and 5 chopsticks

Answer: a


LET'S START!

Questions

A process can be terminated due to
a) normal exit
b) fatal error
c) killed by another process
d) all of the mentioned

Answer : d, divide by zero can be considered as fatal error

What is the ready state of a process?
a) when process is scheduled to run after some execution
b) when process is unable to run until some task has been completed
c) when process is using the CPU
d) none of the mentioned

Answer : a
Explanation : When process is unable to run until some task has been completed, the process is in blocked state and if process is using the CPU, it is in running state.

Questions

What is interprocess communication?

Answer : b

a) communication within the process
b) communication between two process
c) communication between two threads of same process
d) none of the mentioned

A set of processes is deadlock if

a) each process is blocked and will remain so forever
b) each process is terminated
c) all processes are trying to kill each other
d) none of the mentioned

Answer : a

A process stack does not contain

Answer : d

a) function parameters **b) local variables**
c) return addresses **d) PID of child process**

Each request requires that the system consider the _____, _____, _____ to decide whether the current request can be satisfied or must wait to avoid a future possible deadlock. (choose three)

Answer: a, c and d

a) resources currently available
b) processes that have previously been in the system
c) resources currently allocated to each process
d) future requests and releases of each process

Given a priori information about the _____ number of resources of each type that maybe requested for each process, it is possible to construct an algorithm that ensures that the system will never enter a deadlock state.

a) minimum
b) average
c) maximum
d) approximate

Answer: c

Process Synchronization & Deadlock

A deadlock avoidance algorithm dynamically examines the _____, to ensure that a circular wait condition can never exist.

a) resource allocation state
b) system storage state
c) operating system
d) resources

Answer: a

Explanation: Resource allocation states are used to maintain the availability of the already and current available resources.

A state is safe, if :

a) the system does not crash due to deadlock occurrence
b) the system can allocate resources to each process in some order and still avoid a deadlock
c) the state keeps the system protected and safe
d) All of these

Answer: b

Questions & Answers

A system is in a safe state only if there exists a :

a) safe allocation

b) safe resource

c) safe sequence

d) All of these

Answer: c

CPU fetches the instruction from memory according to the value of

a) program counter

b) status register

c) instruction register

d) program status word

Answer : a

Which system call returns the process identifier of a terminated child?

a) wait
b) exit
c) fork
d) get

Answer : a; The **system call wait**() is easy. This **function** blocks the **calling** process until one of its child processes exits or a signal is received. **wait**() takes the address of an integer variable and returns the process ID of the completed process.

The address of the next instruction to be executed by the current process is provided by the

a) CPU registers
b) program counter
c) process stack
d) pipe

Answer : b

General structure of a process consists of

a)Critical section b) remainder section

c) Race condition d) both a) and b)

Scheduling a process from Ready Queue to CPU is done by:

a) Short Term Scheduler

b) Middle Term Scheduler

c) Long Term Scheduler

d) Dispatcher

a)

B.Tech(CSE) - Process

1. A thread is a

a)Task b) process

c) Program d) light weight process

2. Banker's Algorithm for Resource Allocation deals with

a)Deadlock prevention b) deadlock avoidance

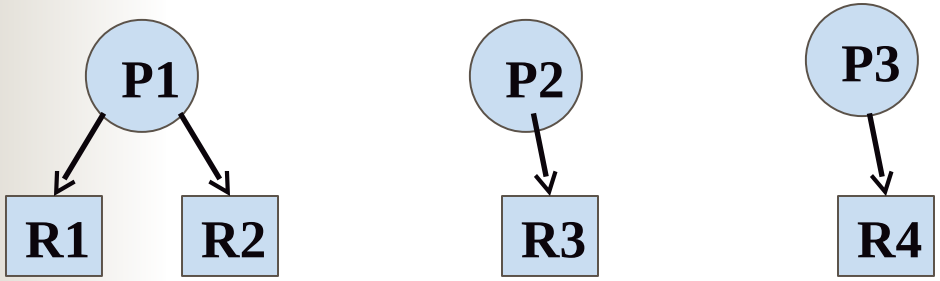
c) Deadlock recovery d) mutual exclusion

b)

3. An OS contains 3 user process each requiring 2 units of resources R. The minimum number of units of R such that no deadlocks will ever arise is:

a) 3 b) 5 c) 4 d) 6

c)



OS questions

Why are the programs and data not resided in main memory permanently?

Answer

There are two reasons:

1) Main memory is usually too small to store all needed programs and. Data permanently.

Main memory is a *volatile* storage device that loses its contents when power is turned off or otherwise lost

What are the activities of the operating system in regard to process management?

Answer

The operating system is responsible for the following activities in regard to process management:

1) Creating and deleting both user and system processes
2) Suspending and resuming processes
3) Providing mechanisms for process synchronization
4) Providing mechanisms for process communication
5) Providing mechanisms for deadlock handling

Explain with diagram the process states?

Answer

Each process may be in one of the following states:

1. New. The process is being created.
2. Running. Instructions are being executed.
3. Waiting. The process is waiting for some event to occur (such as an I/O completion or reception of a signal).
4. Ready. The process is waiting to be assigned to a processor.
5. Terminated. The process has finished execution.

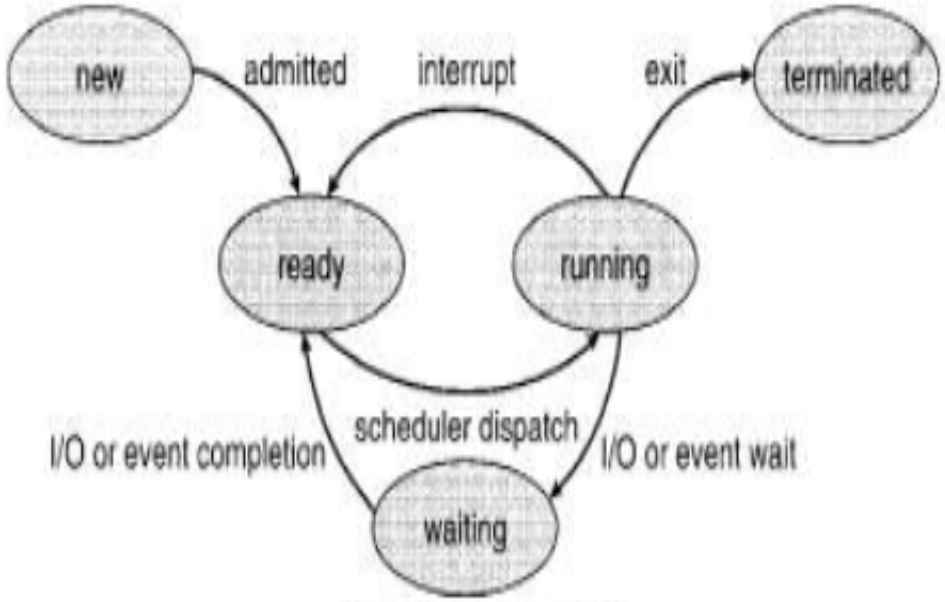


Diagram of process state.

What are the Operations on Processes?

Answer

The operations on processes are process **creation** and **termination.**

1. **Process Creation**: A process may create several new processes, via a create-process system call, during the course of execution. The creating process is called a ***parent process***, and the new processes are called children of that process.
2. **Process Termination**: A process terminates when it finishes executing its final statement and asks the operating system to delete it or by another process via an appropriate system call.

In process creation, what are the possibilities in concerned (1) Parent execution (2) Address space of the new process (child)?

Answer

(1) There are two possibilities of execution:

a) The parent continues to execute concurrently with its children.

b) The parent waits until some or all of its children have terminated.

(2) There are also two possibilities in terms of the address space of the new process:

a) The child process is a duplicate of the parent process .

b) The child process has a new program loaded into it.

A parent may terminate the execution of one of its children. What are the reasons?

Answer

a) The child has exceeded its usage of some of the resources that it has been allocated.
b) The task assigned to the child is no longer required.
c) The parent is exiting, and the operating system does not allow a child to continue if its parent terminates.

Q : How can we avoiding deadlocks occur?

Answer

For avoiding deadlocks is to require additional information about how resources are to be requested.

Each request requires the following:

1) The resource currently available.
2) The resource currently allocated to each process.
3) The future requests and releases of each process.

The above information is used to decide whether the current request can be satisfied or must wait to avoid a possible future deadlock.

A **deadlock-avoidance algorithm** dynamically examines the resource-allocation state to ensure that a circular wait condition can never exist. The resource-allocation *state* is defined by the number of available and allocated resources and the maximum demands of the processes.

OS questions

What are the requirements of solution the critical-section problem?

The requirements are:

1. **Mutual exclusion. If process Pi is executing in its critical section, then no** other processes can be executing in their critical section.
2. **Progress. If no process is executing in its critical section and some** processes wish to enter their critical section, then only those processes that are not executing in their remainder sections can participate in the decision on which will enter its CS next, and this selection cannot be postponed indefinitely. (No process should have to wait forever to enter its critical section.
3. **Bounded waiting. There exists a bound, or limit, on the number of times** that other processes are allowed to enter their critical sections after a process has made a request to enter its critical section and before that request is granted