**Dead lock**

1. **The deadlock**

A deadlock is a situation in which two processes sharing the same resource are effectively preventing each other from accessing the resource, resulting in both programs ceasing to function.

The earliest computer [operating systems](https://www.techtarget.com/whatis/definition/operating-system-OS) ran only one program at a time. All of the resources of the system were available to this one program. Later, operating systems ran multiple programs at once, interleaving them. Programs were required to specify in advance what resources they needed so that they could avoid conflicts with other programs running at the same time.

Here is the simplest example:

Program 1 requests resource 1 and receives it.

Program 2 requests resource 2 and receives it.

Program 1 requests resource 2 and is queued up, pending the release of B.

Program 2 requests resource 1 and is queued up, pending the release of A.

Now neither program can proceed until the other program releases a resource.

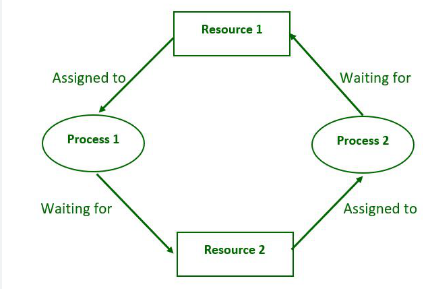


Figure (1) deadlock example

1. **Methods for Handling Deadlocks**

* Generally speaking there are three ways of handling deadlocks:
  1. **Deadlock prevention or avoidance** - Do not allow the system to get into a deadlocked state.
  2. **Deadlock detection and recovery** - Abort a process or preempt some resources when deadlocks are detected.
  3. **Ignore the problem all together** - If deadlocks only occur once a year or so, it may be better to simply let them happen and reboot as necessary than to incur the constant overhead and system performance penalties associated with deadlock prevention or detection