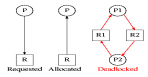
To define deadlock, first I would define process.

Process : As we know process is nothing but a `program` in execution.

Resource : To execute a program process needs some resources. Resource categories may include memory, printers, CPUs, open files, tape drives, CD-ROMS, etc.

Deadlock : Deadlock is a situation or condition when two or more processes are holding some resources and trying to acquire some more resources, and they can not release the resources until they finish there execution.

Deadlock condition or situation



In the above diagram there are two process **P1** and **p2** and there are two resources **R1** and **R2**.

Resource **R1** is allocated to process **P1** and resource **R2** is allocated to process **p2**. To complete execution of process **P1** needs resource **R2**, so **P1** request for **R2**, but **R2** is already allocated to **P2**.

In the same way Process **P2** to complete its execution needs **R1**, but **R1** is already allocated to **P1**.

both the processes can not release their resource until and unless they complete their execution. So both are waiting for another resources and they will wait forever. So this is a **DEADLOCK** Condition.

In order for deadlock to occur, four conditions must be true.

1. **Mutual exclusion** - Each resource is either currently allocated to exactly one process or it is available. (Two processes cannot simultaneously control the same resource or be in their critical section).
2. **Hold and Wait** - processes currently holding resources can request new resources.
3. **No preemption** - Once a process holds a resource, it cannot be taken away by another process or the kernel.
4. **Circular wait** - Each process is waiting to obtain a resource which is held by another process.

and all these condition are satisfied in above diagram.