1. Describe each of the following in **detail**:
2. Deadlock

**Deadlocks:** When jobs are put on hold because they all are waiting on vital resources to be available. Jobs can’t be finished because jobs are all holding each other’s needed resources. These affect multiple jobs.

1. Livelock

**Livelock:** Almost identical to a Deadlock, except that a Livelock has jobs that are constantly shifting resources but not making any actual progress. For example, jobs might keep giving up 1 MB of resources simultaneously which only lands both jobs right back where they started. More liberal algorithms can cause this and make the system confused.

1. Starvation

**Starvation:** When resources are being constantly denied to processes that need them to get finished. Conservative algorithms typically cause this as they allow processes to hoard resources for so long rather than allocate them fairly.

1. What is the primary difference between Deadlock and Livelock?

Deadlocks are when system resources are all halted and not being moved, Livelocks are constantly shifting the disputed resources but to no real conclusion.

1. How many **cases** of Deadlock are described in the textbook?

**Seven cases** are described, which are:

* + File Request Deadlocks
  + Databases Deadlocks
  + Dedicated Device Allocation Deadlocks
  + Multiple Device Allocation Deadlocks
  + Spooling Deadlocks
  + Network Deadlocks
  + Disk Sharing Deadlocks

1. Describe the four (4) **conditions** of Deadlock or Livelock.

* **Mutual Exclusion:** The condition in which a process holds onto a dedicated resource and preventing another process from using it to finish.
* **Resource Holding:** When processes hold onto their resources, do not share any of them and wait for other processes to give up theirs.
* **No preemption:** The lack of temporary reallocations for resources. Resources are dedicated to one or more processes.
* **Circular Wait:** When processes are waiting for each other to voluntarily release a resource so one may continue first.

Only one of these conditions needs to be dealt with to prevent deadlocks.

1. Answer the following questions about diagramming Deadlocks:
2. What symbol is used to represent a resource?

**Squares**

1. What symbol is used to represent a process?

**Circles**

1. What does a dashed line represent?

**A process is waiting on a resource.**

1. What does a solid line represent?

**A resource has been allocated to a process.**

1. Describe each of the following Deadlock **strategies** in **detail**:
   1. Prevention

This strategy deals with deadlocks by making sure the four conditions for deadlocks can’t happen. We can prevent the possibility of no preemption by allowing the OS to deallocate resources. However, conditions cannot be eliminated from every resource which can make this strategy a problem.

* 1. Avoidance

The OS won’t attempt to remove the possibility of one of the conditions but try to avoid the condition if it is perceived to become a problem in the future.

* 1. Detection

Using directed resource graphs, we can detect deadlocks. Instead of running every time there’s a request like with avoidance, an algorithm can detect circularity.

* 1. Recovery

The OS either terminates a victim of a deadlock or takes resources from a process that isn’t deadlock and uses them to accommodate a deadlocked job.

1. Answer the following questions about the Prevention, Avoidance, and Detection Deadlock strategies:
   1. Which is the easiest to implement?

**Prevention**

* 1. Which is the most efficient to to use?
  2. Which is the least efficient to to use?

|  |  |  |  |
| --- | --- | --- | --- |
| **Job #** | **Devices** **Allocated** | **Maximum** **Required** | **Remaining** **Needs** |
| **1** | **5** | **6** | 1 |
| **2** | **4** | **7** | 3 |
| **3** | **2** | **6** | 4 |
| **4** | **0** | **2** | 2 |
| **Total Devices Allocated:** | | | **11** |
| **Total Devices in the System:** | | | **12** |

1. Refer to the table above to answer the following questions.  
   1. What are the **Remaining Needs** of each of the four (4) jobs?

* Job 1: **1**
* Job 2: **3**
* Job 3: **4**
* Job 4: **2** 
  1. How **many** devices remain **available** for assignment to jobs?

Total Devices in System (12) – Total Devices Allocated (11) = **1 Device remains**

* 1. Is the system currently in a Safe or Not Safe state?

Because Job 1 currently only needs 1 more device, and we have 1 device remaining, the system is **safe**.

* 1. If the system is in a Safe state, which **job** would need to request its remaining devices and run to completion **first**?

Since the job is in a safe state, **job 1** is the job that would request the remaining devices and run to completion first.

* 1. If the system is in a Not Safe state, which **job** would be the **best** choice for **termination**?

**The job is in a safe state**, so this doesn’t apply.

|  |  |  |  |
| --- | --- | --- | --- |
| **Job #** | **Devices** **Allocated** | **Maximum** **Required** | **Remaining** **Needs** |
| **1** | **5** | **8** | 3 |
| **2** | **3** | **9** | 6 |
| **3** | **4** | **8** | 4 |
| **Total Devices Allocated:** | | | 12 |
| **Total Devices in the System:** | | | **14** |

1. Refer to the table above to answer the following questions.  
   1. What are the **Remaining Needs** of each of the three (3) jobs?

* Job 1: **3**
* Job 2: **6**
* Job 3: **4**
  1. How **many** devices remain **available** for assignment to jobs?

Total Devices in System (14) – Total Devices Allocated (12) = **2 Device remains**

* 1. Is the system currently in a Safe or Not Safe state?

Since the only 2 devices that remain aren’t enough for any of the jobs requesting devices, this system is **not safe**.

* 1. If the system is in a Safe state, which **job** would need to request its remaining devices and run to completion **first**?

**This system is not safe**, so this doesn’t apply.

* 1. If the system is in a Not Safe state, which **job** would be the **best** choice for **termination**?

Since Job 2 has the most devices needed, **job 2 would be the best choice for termination.**

1. Answer the following questions about **Starvation**.
   1. Explain what it means for a job to experience Starvation.

Conservative allocation of resources can cause Starvation. This is where resources are waiting for resources that are unavailable to the point they cannot be executed.

* 1. How is starvation similar to or different from **Deadlock** or **Livelock**?

Starvation causes certain processes to be put in a stasis as resources cannot be allocated. This is just like Deadlocks or Livelocks but these can affect the *entire system*. Starvation typically affects *certain* processes.

* 1. What can an operating system do to prevent a starving job from "remaining in the system forever"?

Algorithms that can simply detect waiting times and forcefully allocate the resources to jobs that have been waiting a long time.

1. Using the six (6) Recovery methods discussed on pages 160 and 161, create a table with the following information (assume a **total** of seven (7) jobs are in the system and three (3) of the jobs are **deadlocked**):

* Method 1 – Termination - Terminate all jobs that are active and restart from the beginning. (Number of victims: ?)
* Method 2 – Termination – Terminate jobs only involved in the deadlock and ask users to resubmit them. (Number of victims: ?)
* Method 3 – Termination – Terminates jobs involved in the deadlock one at a time and check if the deadlock is eliminated each run. (Number of victims: ?)
* Method 4 – Preemption – Jobs maintain a record of progress with a snapshot so they can be interrupted and then be continued later.
* Method 5 – Preemption – Selects a nondeadlocked job and reallocates resources from it to a deadlocked job to break the deadlock. (Number of victims: ?)
* Method 6 – Preemption – Prevents new jobs from entering the system and allows current jobs to run to completion. (Number of victims: ?)