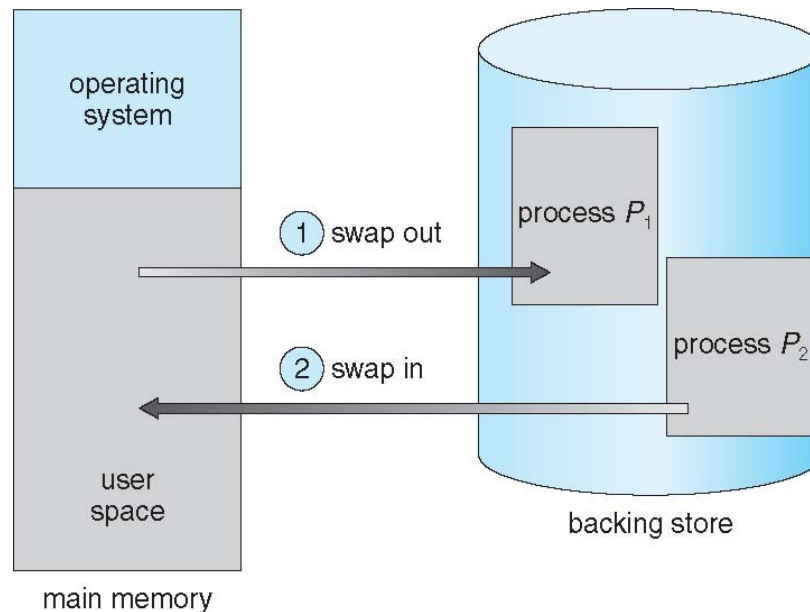# Memory Management

How OS deals with the apps
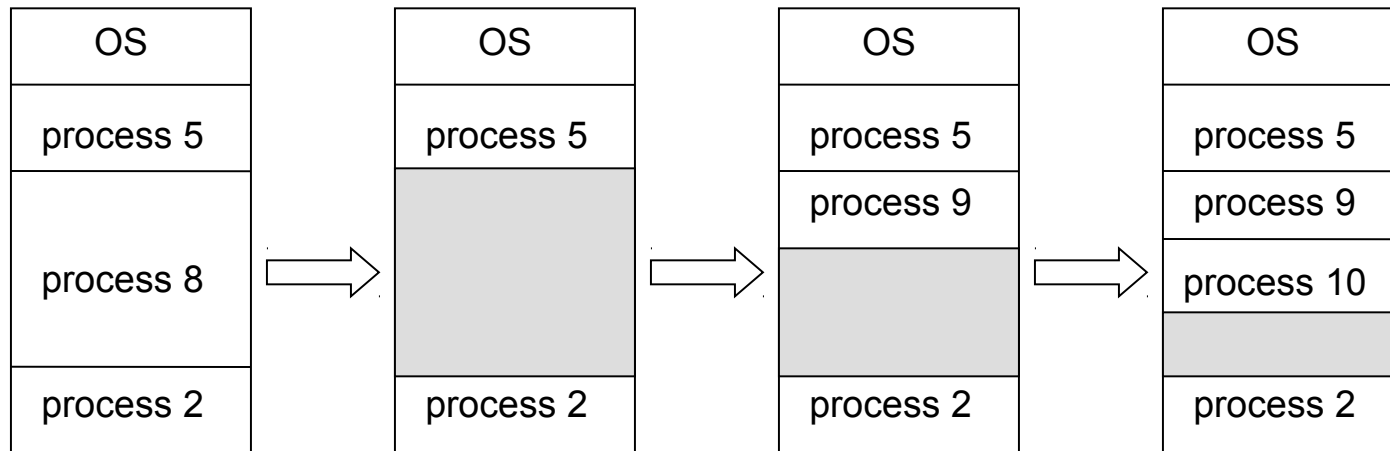
Operating Systems

- "Middleman" to run your executable codes on the computers.

- Need to put your programs into the physical memory space before they are run by CPU.

How to manage them?

- The first approach: simply allocate them based on the size of program code (called "process" in OS)

- Eventually see "holes" scattered in different places

OS
process 5
process 8
process 2

→

OS
process 5
process 2

→

OS
process 5
process 9
process 2

→

OS
process 5
process 9
process 10
process 2

Paging System

- Divide the physical memory space into pages.

  – What is the optimal page size?

- To run a program of size with n pages, OS needs to find the free space of n contiguous pages.

- The OS needs to find a "hole" to put the newly-requested process there.

  – There can be multiple choices.

Which "hole" to fill-in?

- ***First-fit Strategy***: Allocate the FIRST hole that is big enough (reason: simple)

- ***Best-Fit Strategy***: Allocate the SMALLEST hole that is big enough

  - (reason: produce the smallest left-over hole)

- ***Worst-Fit Strategy***: Allocate the LARGEST hole

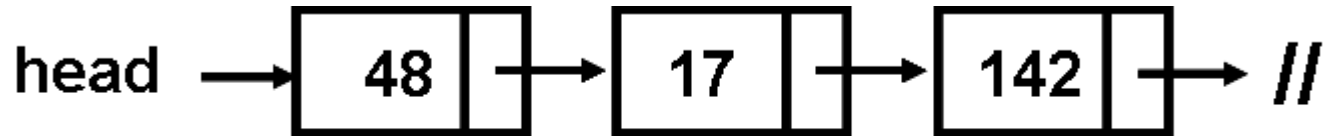 - (reason: produce the largest left-over hole)

Fragmentations

- ***External Fragmentation***: Total free-memory exists to satisfy a request, but it is not contiguous

  - (solution: compact the free memory into one contiguous space)

- ***Internal Fragmentation***: Part of the nature in the paging system

  - (solution: use smaller page size)

HOW to Manage?

- ***Linked Lists***: keep track of <u>allocated</u> and <u>free</u> memory pages

  - When a request (called "job" in OS) comes, use the specific strategy to find the "hole"

  - Allocate the free-memory pages to the allocated-memory pages

  - Update BOTH linked lists

  - Calculate the fragmentation
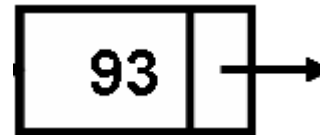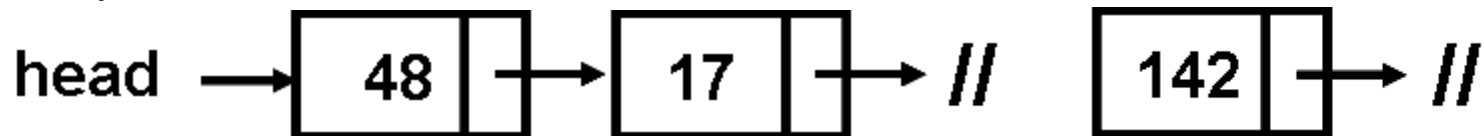
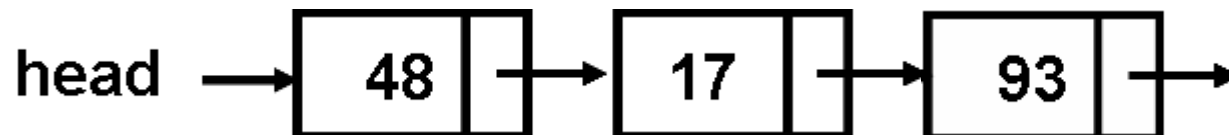 - Do the compaction if necessary

Linked list: Addition

head → [48 |] → [17 |] → [142 |] → //

Step 1   Step 2

[|] →   [93 |] →

Step 3

head → [48 |] → [17 |] → //   [142 |] → //

Step 4

head → [48 |] → [17 |] → [93 |] →
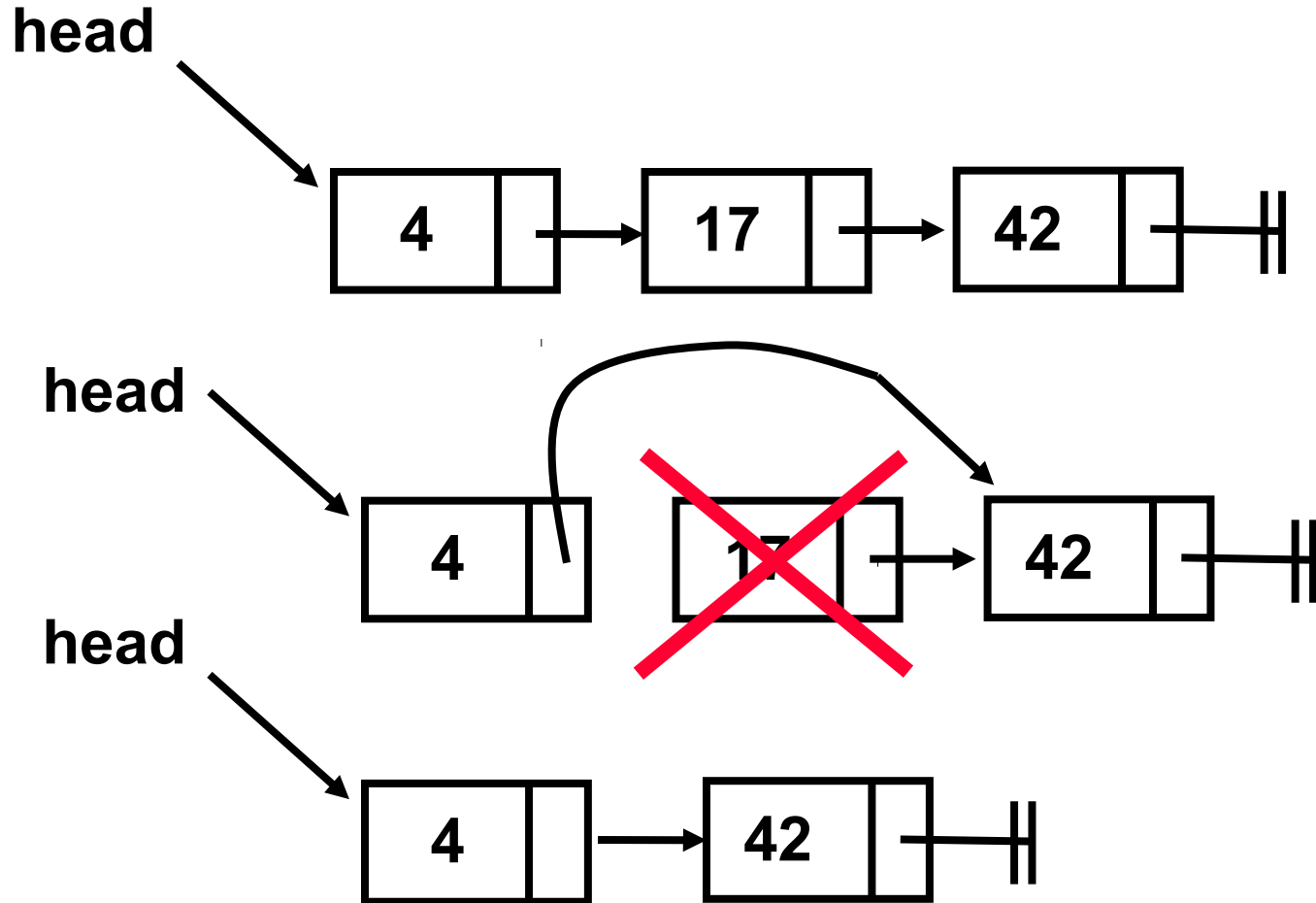
Linked list: Deletion

More Thinking …

- ***Other operations***: traverse, split, merge, etc …

- ***Doubly linked lists***: to speed up the search

- ***Logical paging and page Table***: to increase the flexibility on the contiguous space

- ***More areas to apply***: disk de-fragmentation and "junk-file cleanup" and "memory boost"