- Partition gets created based on the requirement of the incoming process

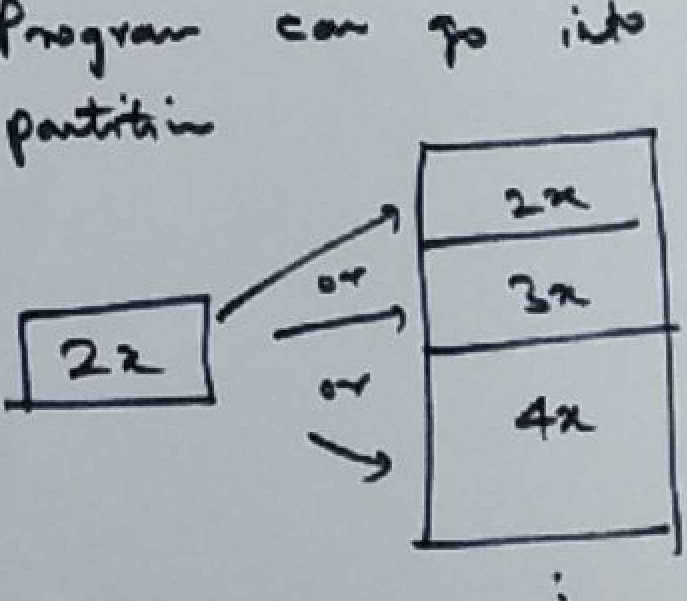
Issues in Partitioning

— Placement Algos —

— Memory Wasted. —
(Fragmentation)

Problem

— Program can go into multiple partitions



Placement Algorithm

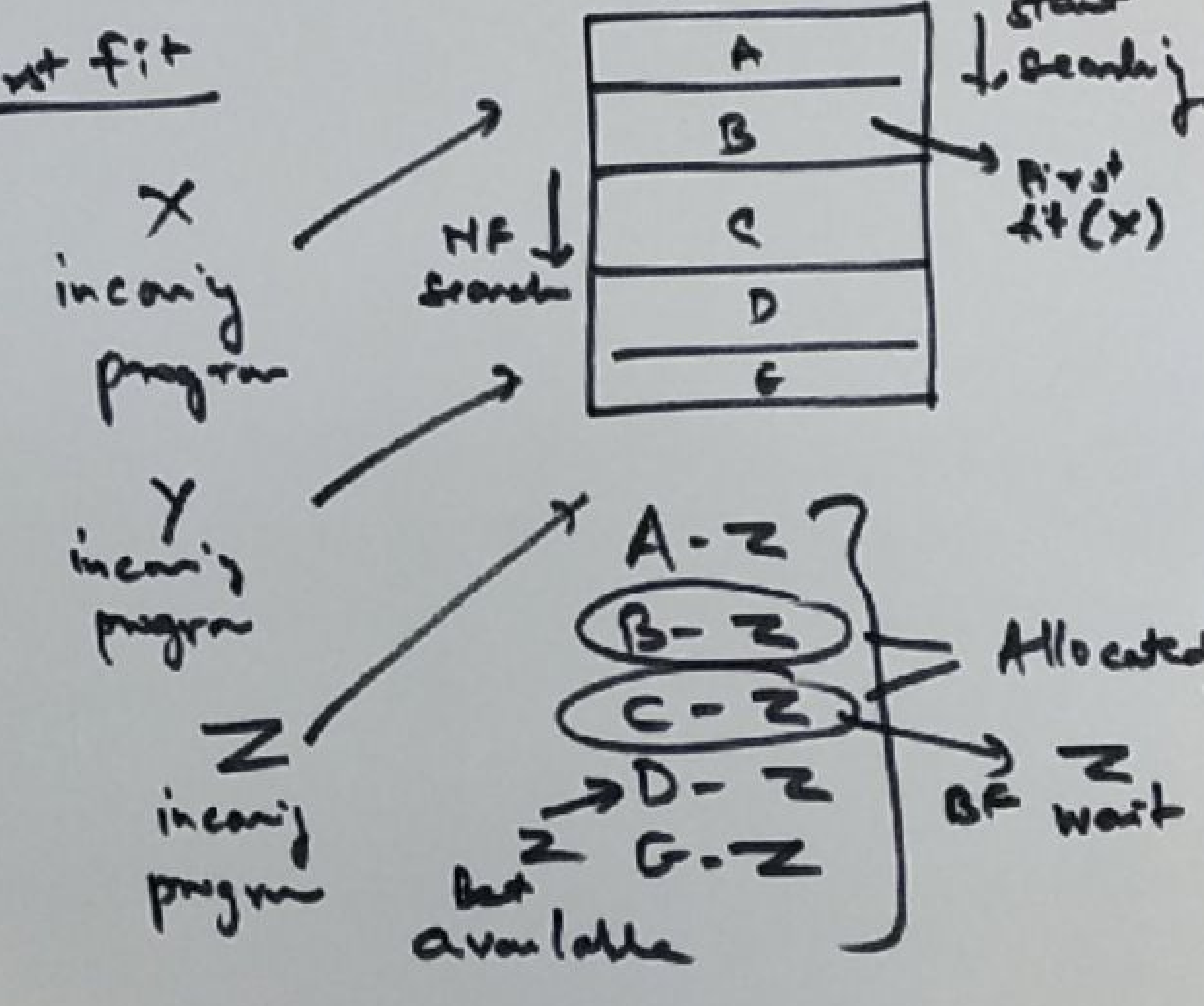
① First-fit — from beginning

② Best-fit — Minimum wastage of memory
 └Best possible fit

③ Worst-fit — Maximum wastage of memory

④ Next-fit → starts from point of last allocation

First fit



Best possible

Best Available (Sub-optimal) || different than Bestfit (optimal)

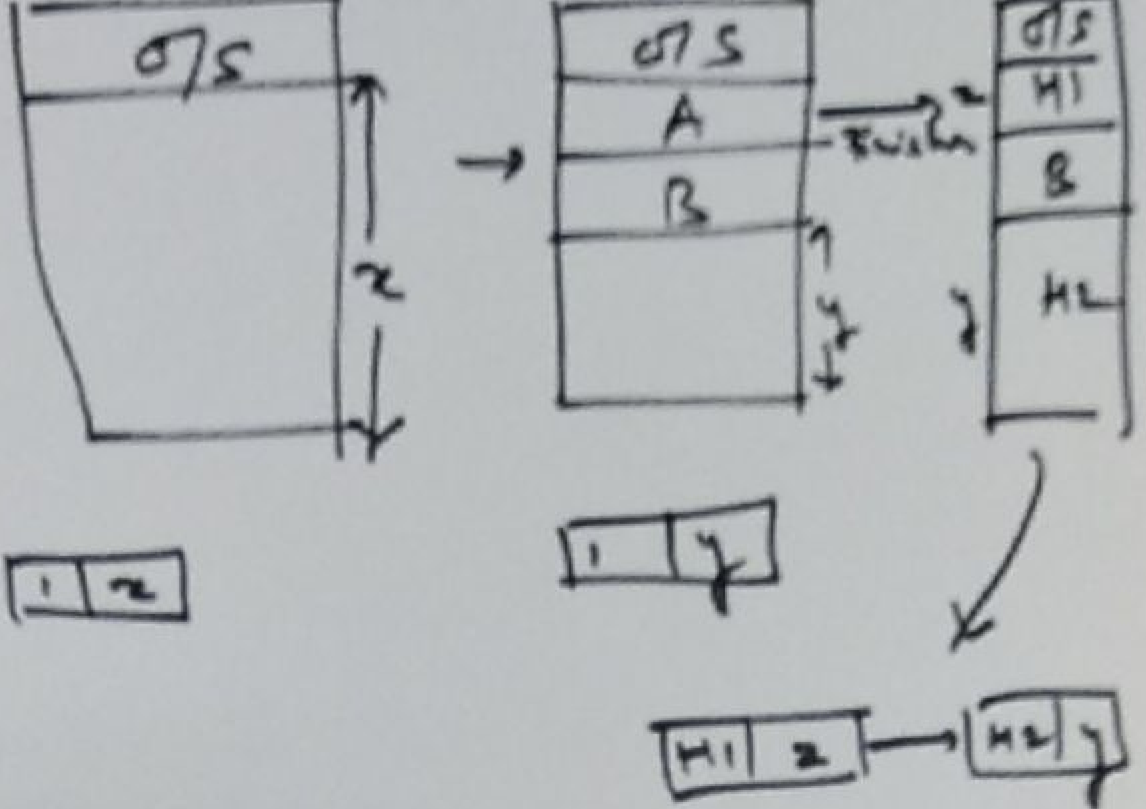
Worst fit → largest space

Variable Partitioning

└ Hole (Free space) → Every memory allocation creates a partition & a hole

$$H \xrightarrow[A]{\text{Allocate}} \left[\begin{matrix} A \\ H-A \end{matrix}\right]$$

— Hole list — scattered across memory



Placement Algo → Fixed Partitioning (Partition)

→ Variable Partitioning (Hole)

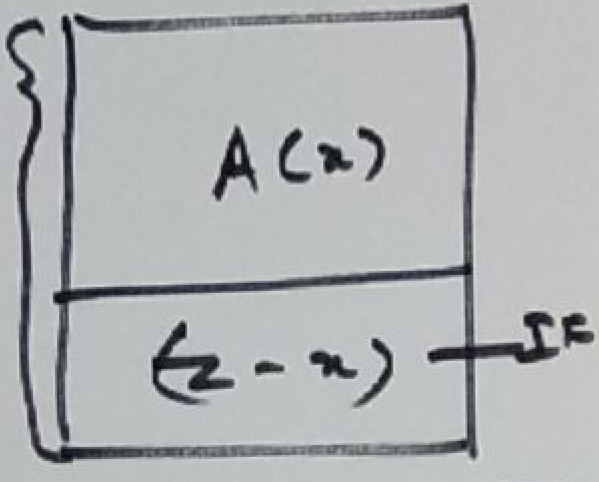
Memory fragmentation

① Internal fragmentation (IF)

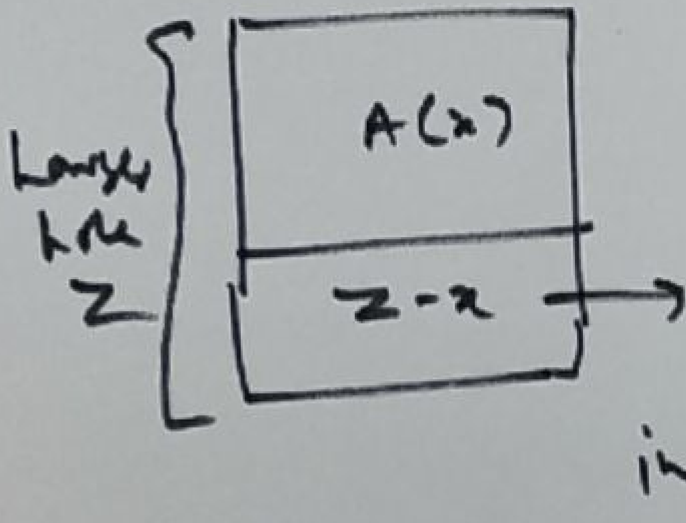
(inside a partition/page..
memory)
block

A
(size = x)

(Z > x)



Partition size = Z
(Fixed partitioning)



becomes a
new hole
in Variable partition

— All memory partition are created to the size of powers of 2 (closest)

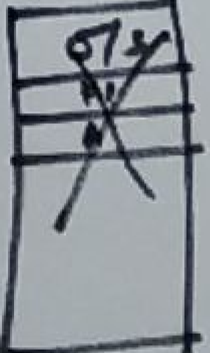
A
(20K) 3 bytes ⟶ $\boxed{\text{4 bytes}}$

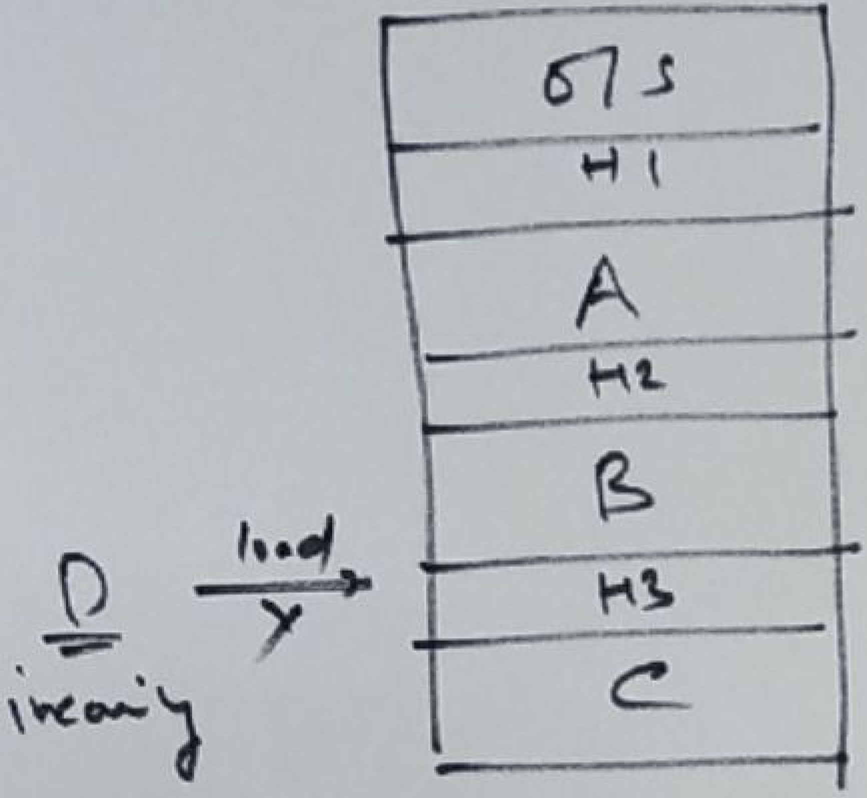
5bytes ⟶ $\boxed{\text{8bytes}}$

Theory ⟶ No IF

Implementation ⟶ Yes IF
(less → high)

② External fragmentation
(outside the partition)





$D > H1$

$D > H2$

$D > H3$

H1, H2, H3 are scattered

Contiguous (compaction)

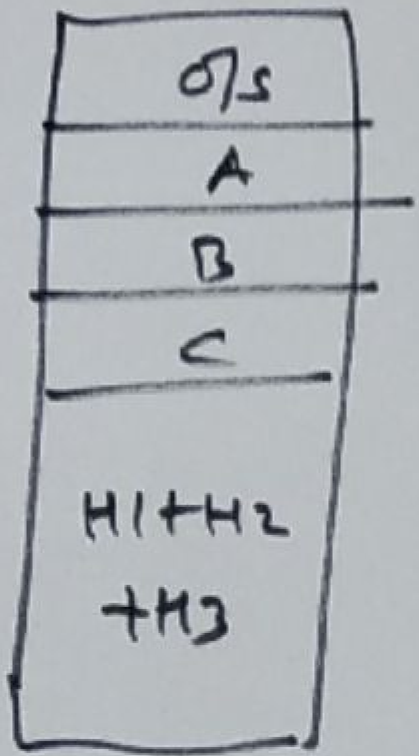
$D < H1+H2$

$D < H2+H3$

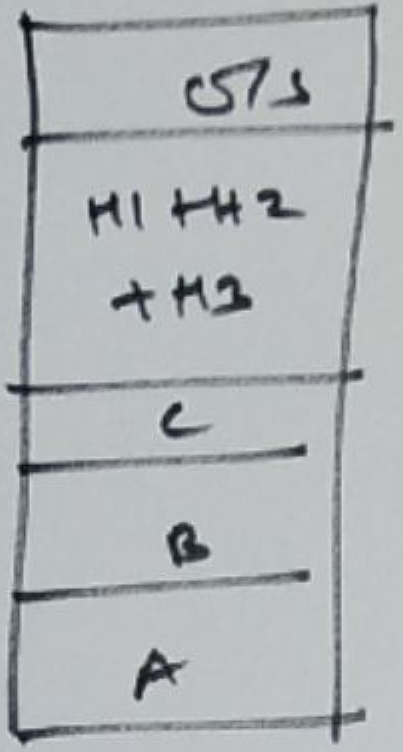
$D < H1+H3$

$D < \boxed{H1+H2+H3}$ EF

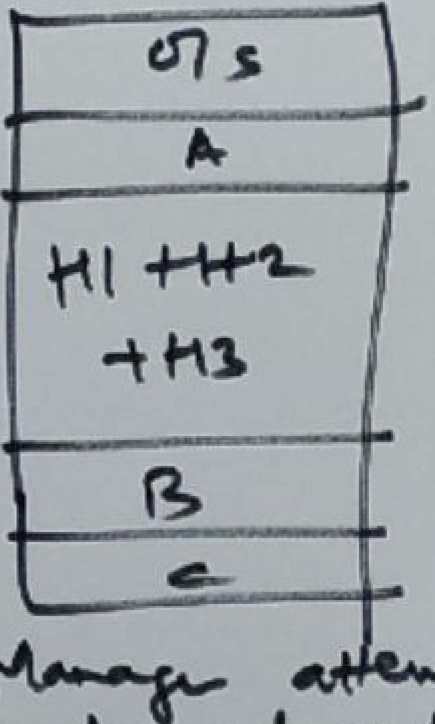
No EF $\boxed{D > H1+H2+H3}$



or



or



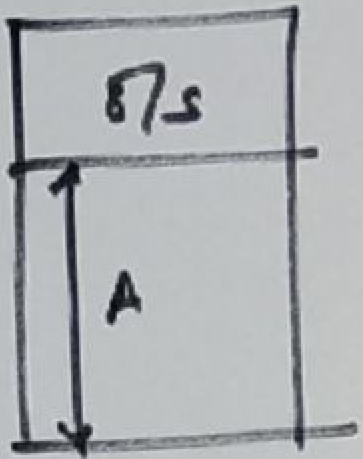
Compaction
↓
Solution
to
EF
(in many
cases)

— Memory Manager attempt to optimize the relocation effort

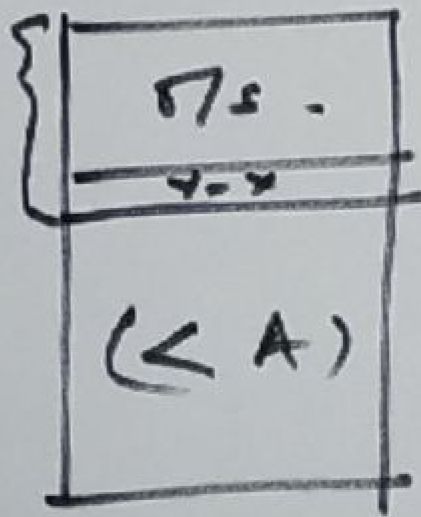
3 Table fragmentation (TF)

- Data structures related to memory management
- Used

Memory Management scheme 1 (TF = X) — Data structure X



Memory Management Schem 2 (TF = Y) — Data structure Y



$(Y >> X)$

$(Y - X)$ extra.

Total frag = $\underline{IF + EF} + TF$