Allocation Methods in contiguous

Introduction: In continuous memory management there are 4 algorithms. Through these algorithms we can allocate the upcoming process in proper hole to avoid from internal fragmentation. These algorithms are known as Partition Allocation Methods In Contiguous.

1. First Fit: In this algorithm OS allocate the process to first Hole that is big enough.

- It is very fast and easy to implement
- First fit algorithm starts scanning the partitions from the beginning of memory every time to load new process in main memory.
- When it found first big enough partition, that can accommodate the loading process, then scanning stop and process loaded successfully.

2. Next Fit: Next fit is modified of first fit but it will search for the first big enough partition from the last allocation partition.

3. Best Fit: In this algorithm OS allocate the process to that partition which is first smallest enough (among the all free available partition) to accommodate that process.

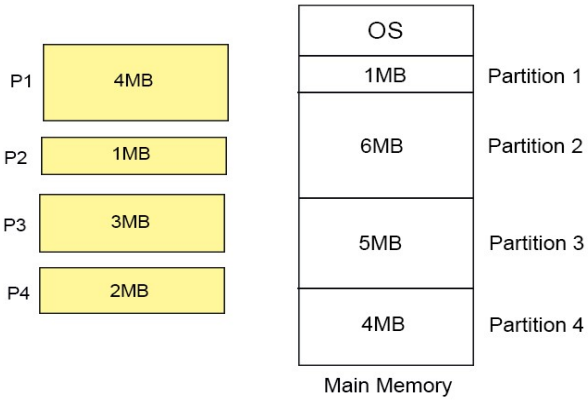
- It is time consuming algorithm but internal fragmentation factor is minimized.

4. Worst Fit: In this algorithm OS allocate the process to that partition which is first Largest enough (among the all free available partition) to accommodate that process.

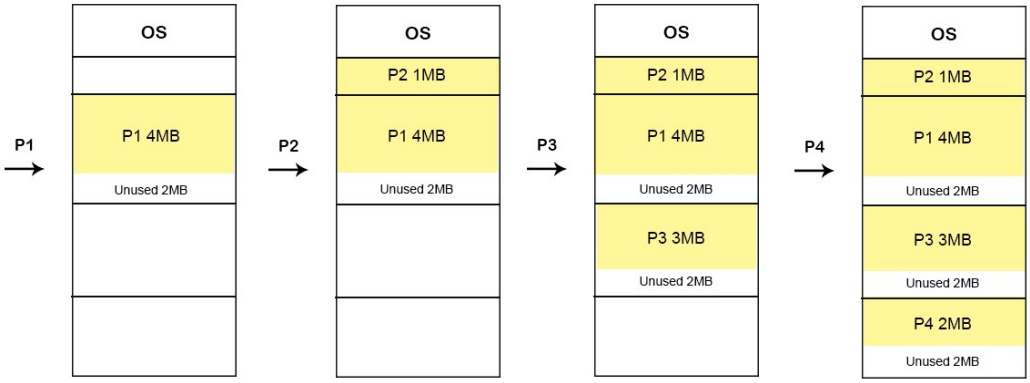
Allocation Methods in FIXED and Dynamic Partitions explained under,

1. Fix Sized Partitions

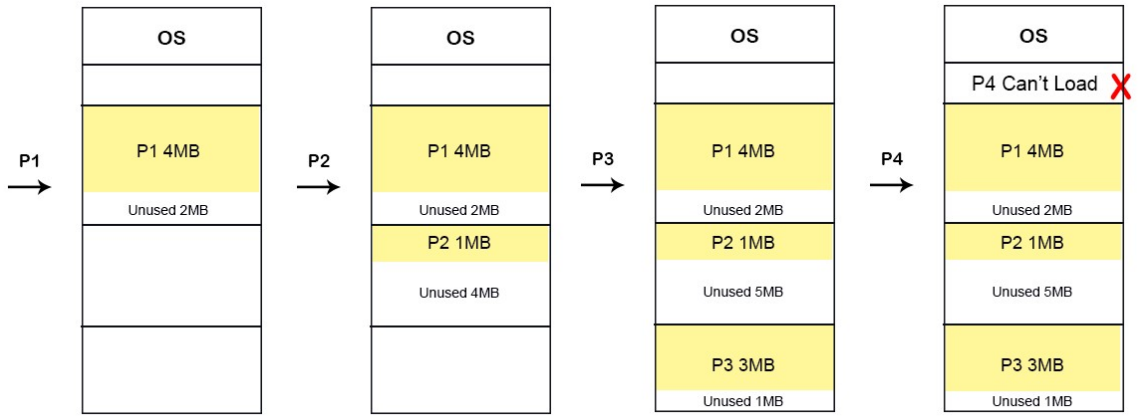
Eg: Let suppose we have four FIX sized partitions with size 1MB, 6MB, 5MB, and 4MB respectively and four incoming processes (P1 to P4) with sizes 4MB, 1MB, 3MB and 2MB respectively.



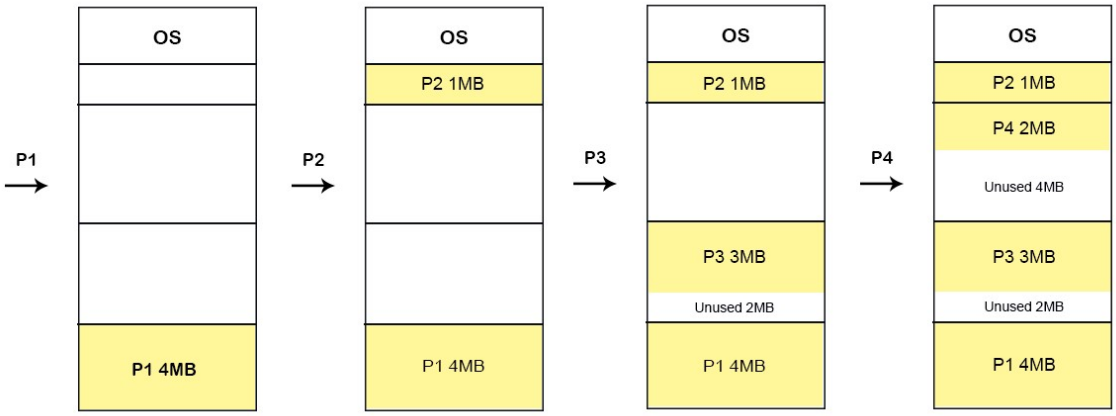
Let's start the First Fit algorithm execution from above mentioned processes and memory.



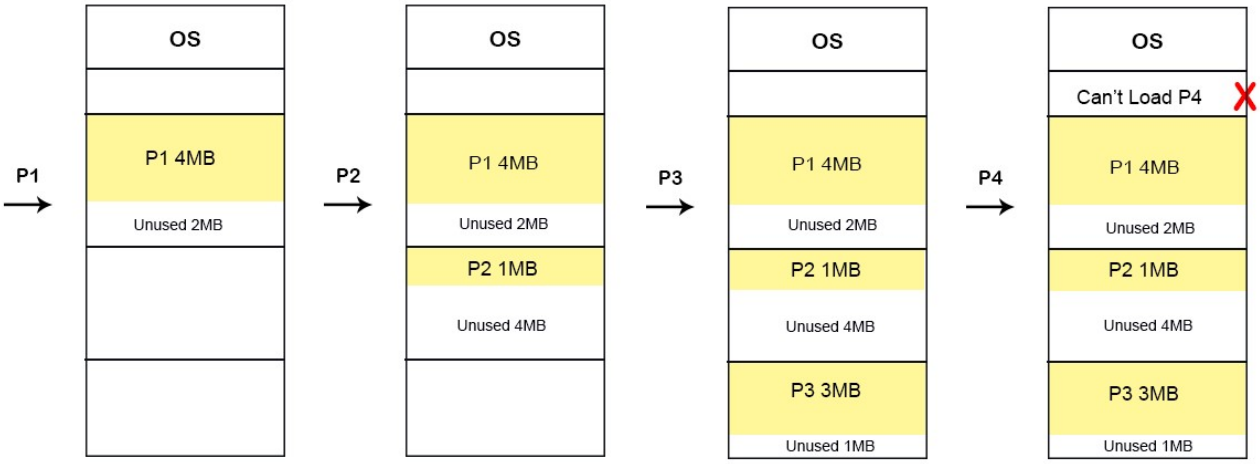
Let's start the Next Fit algorithm execution from above mentioned processes and memory.



Let's start the Best Fit algorithm execution from above mentioned processes and memory.



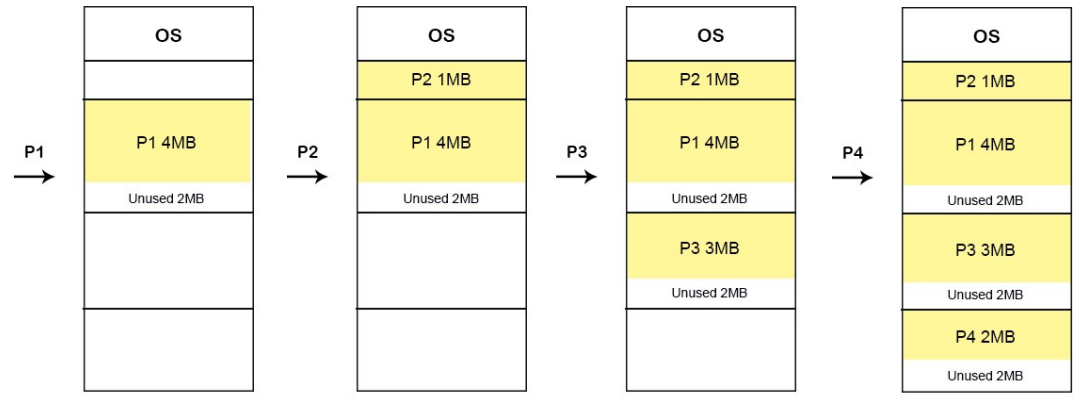
Let's start the Worst Fit algorithm execution from above mentioned processes and memory.



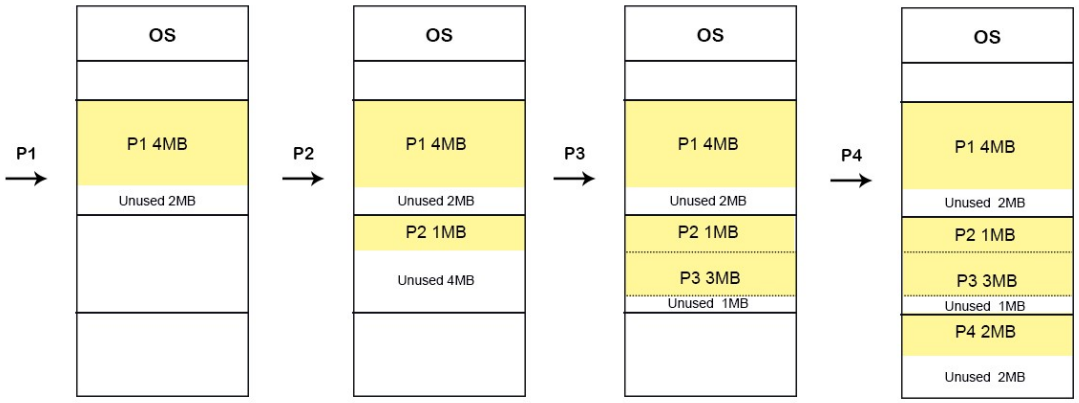
2. Dynamic Partitions

Explain With Example: Let suppose we have four empty Dynamic sized partitions with size 1MB, 6MB, 5MB, and 4MB respectively and four incoming processes (P1 to P4) with sizes 4MB, 1MB, 3MB and 2MB respectively.

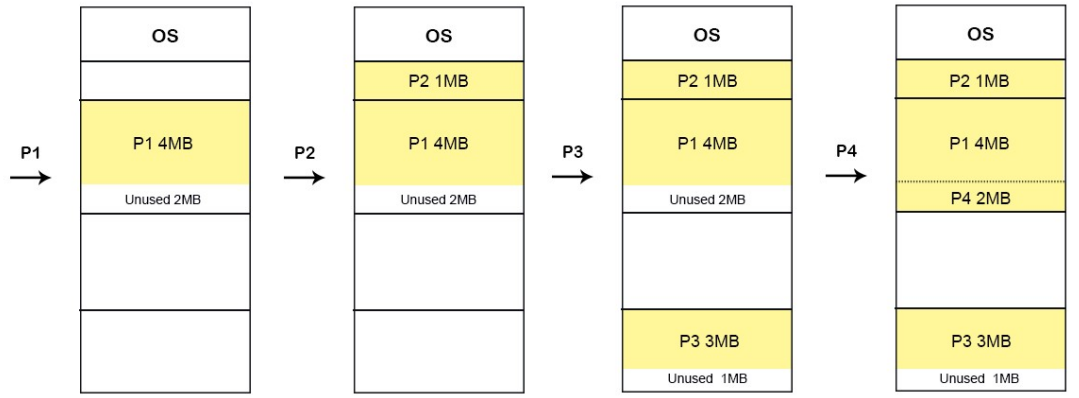
Case 01: First Fit



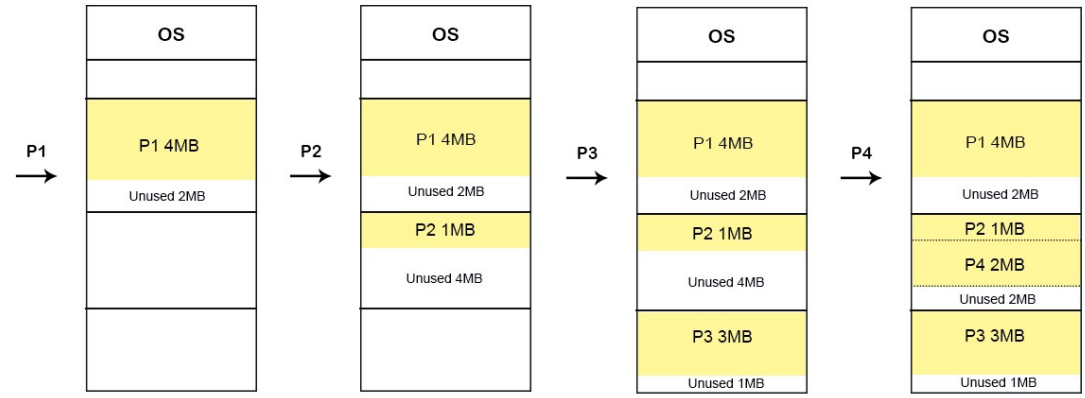
Case 02: Next Fit



Case 03: Best Fit



Case 04: Worst Fit



Important Points-

Point-01:For static partitioning,

- Best Fit Algorithm works best.
- This is because space left after the allocation inside the partition is of very small size.
- Thus, internal fragmentation is least.

Point-02: For static partitioning,

- Worst Fit Algorithm works worst.
- This is because space left after the allocation inside the partition is of very large size.
- Thus, internal fragmentation is maximum.

Point-03: For dynamic partitioning,

- Worst Fit Algorithm works best.
- This is because space left after allocation inside the partition is of large size.
  - There is a high probability that this space might suit the requirement of arriving processes.

Point-04:For dynamic partitioning,

- Best Fit Algorithm works worst.
- This is because space left after allocation inside the partition is of very small size.
- There is a low probability that this space might suit the requirement of arriving processes.

Summary:

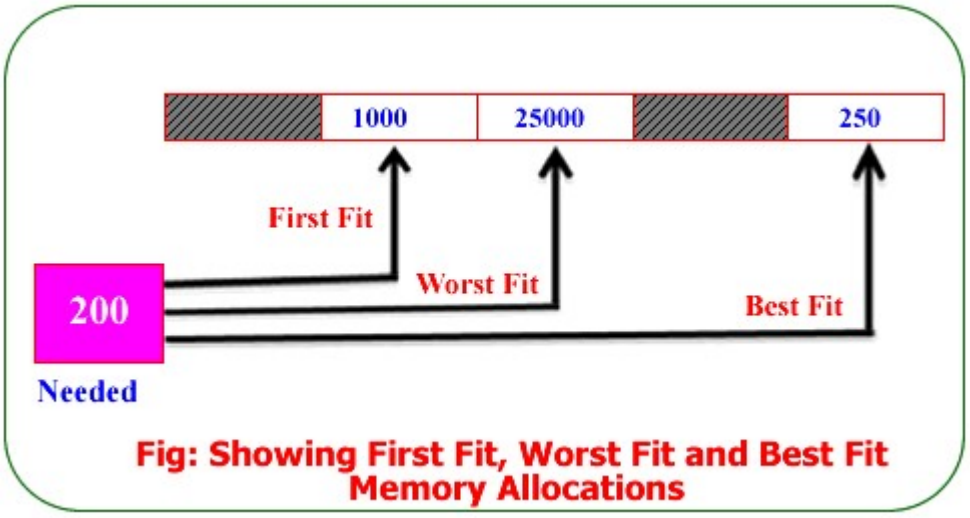


Fig: Showing First Fit, Worst Fit and Best Fit Memory Allocations

Practice Questions

Problem1: In dynamic partitions, If the processes requests are in given order

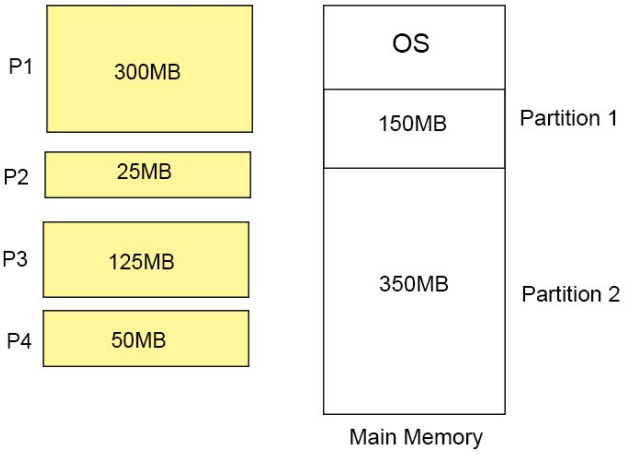
300MB
25MB
125MB
50MB

and two memory blocks are available of size 150MB and 350MB.

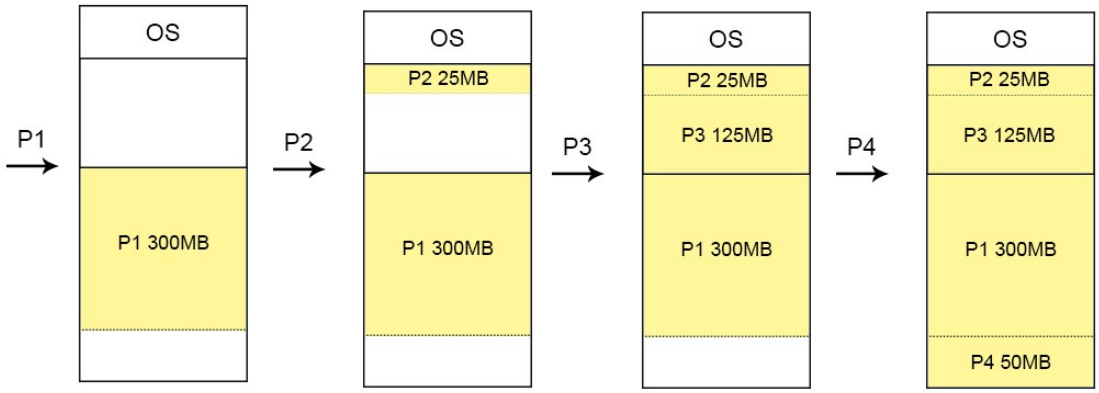
Which of the following partition allocation method is fulfill the above conditions?

A) Best fit but not first fit.
B) Both first fit & best fit.
C) First fit but not best fit.
D) Neither first fit nor best fit.

Solution

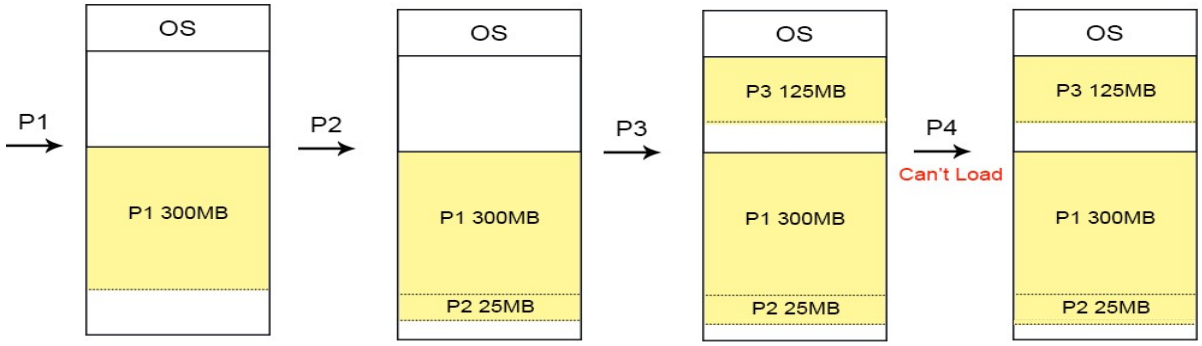


According to First Fit



First Fit

According to Best Fit



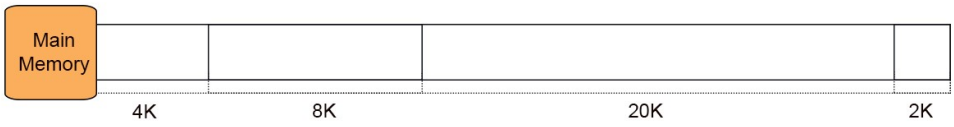
Best Fit

So option C is the correct choice.

Problem2: Question 1: if there are 8 jobs (J1 to J8) arrive at time zero having job sizes 2K, 14K, 3K, 6K, 6K, 10K, 7K, 20K respectively and there usage time 4, 10, 2, 8, 4, 1, 8, 6 respectively then calculate the time at which Job 7 will completed by applying BEST FIT algorithm.

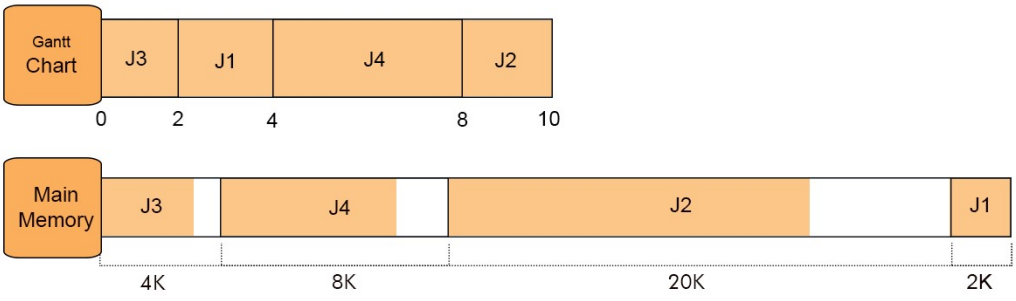
Solution

Process Request For Jobs	J1	J2	J3	J4	J5	J6	J7	J8
Request / Job Size	2K	14K	3K	6K	6K	10K	7K	20K
Usage Time in Memory	4	10	2	8	4	1	8	6



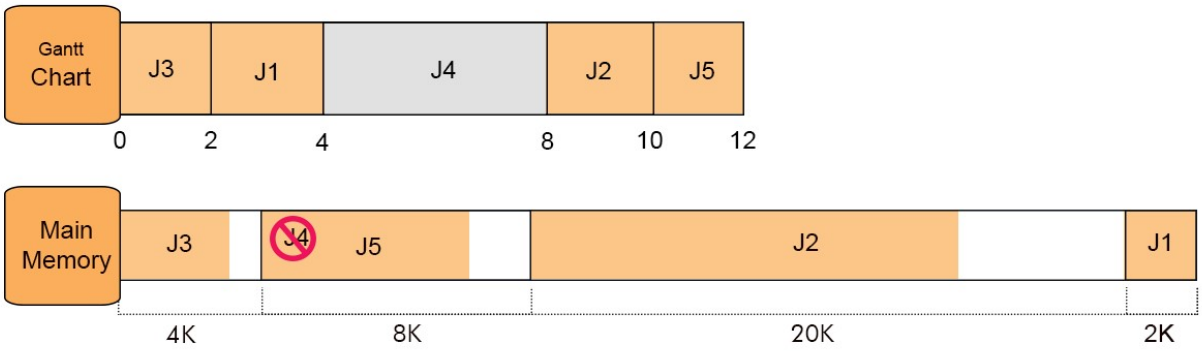
Step 1

As it is best fit case, so J1 fit in 2K, J2 in 20K, J3 in 4K and J4 in 8K memory Partitions. Timeline of completion time of all jobs given below



Step 2

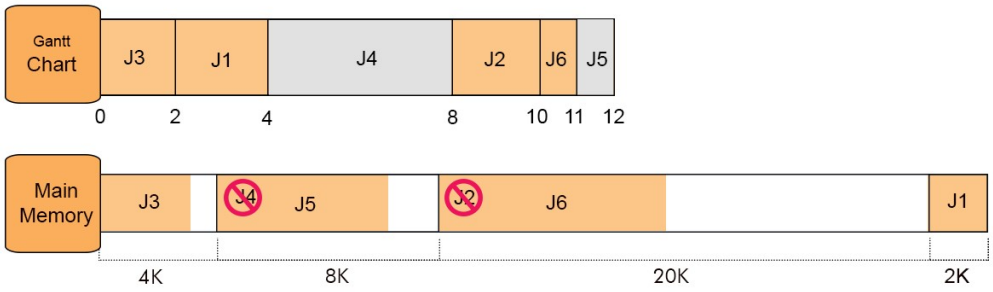
J5 came now, but the memory is already full. J5 wait for termination of any Job so J5 will check the timeline to find his Place in memory. At Point 2 Job J3 terminate but its space is not enough to accommodate J6 size and the same case with J1 at point 4.



So at point 8 Job J4 is completed and J5 enter at point 8 and completed at point 12.

Step 3

J6 came now, but the memory is already full. J6 wait for termination of any Job so J6 will check the timeline to find his Place in memory. At Point 2 Job J3 terminate but its space is not enough to accommodate J6 size and the same case with J1 at point 4.



So at point 10 Job J2 is completed and J6 enter at point 8 and completed at point 11.
Because its execution time is 1.

Step 4

J7 came now, follow the previous rules, so it will replace J6 and terminate at point 19.