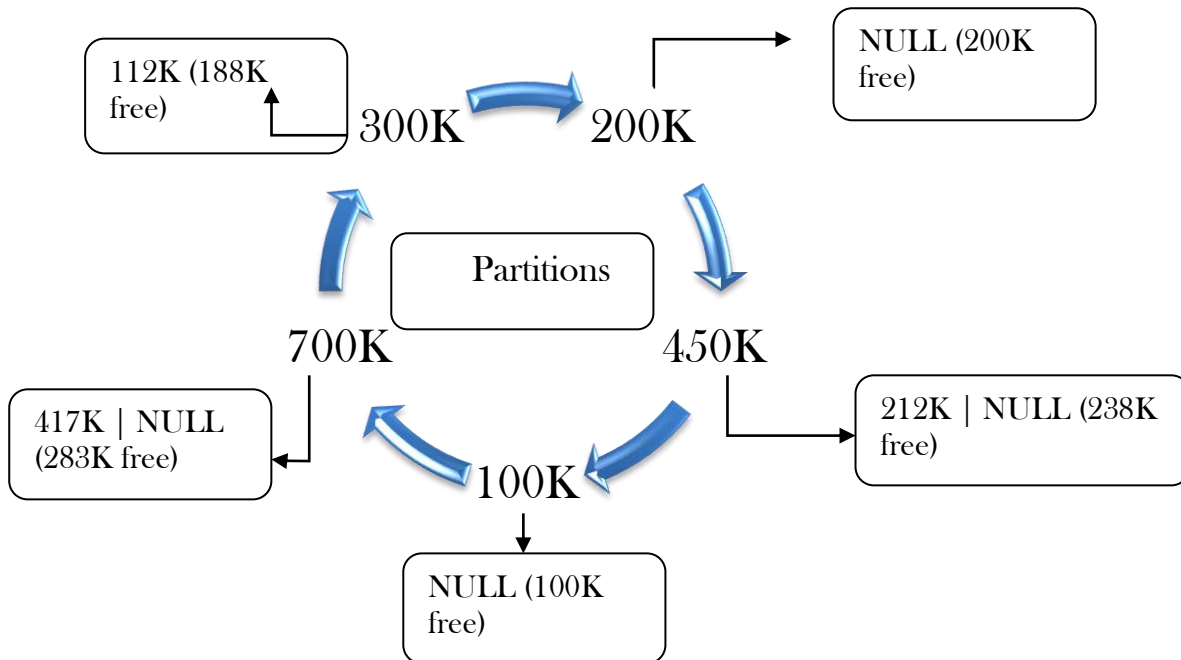
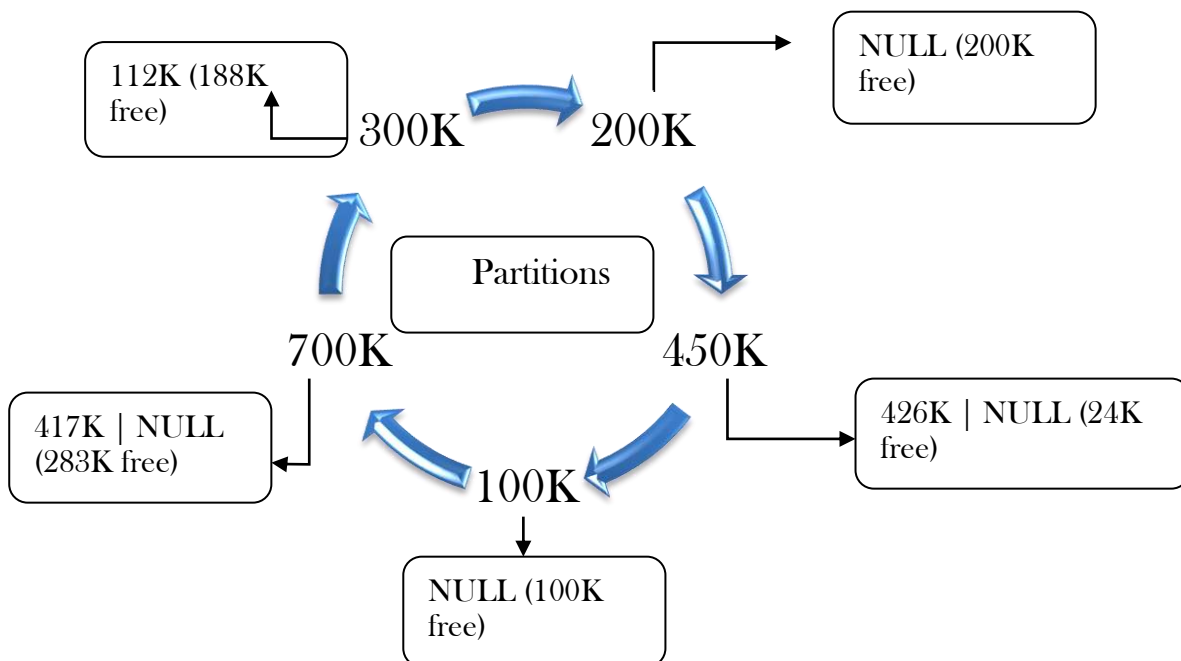


For the First fit, we have the following algorithm:

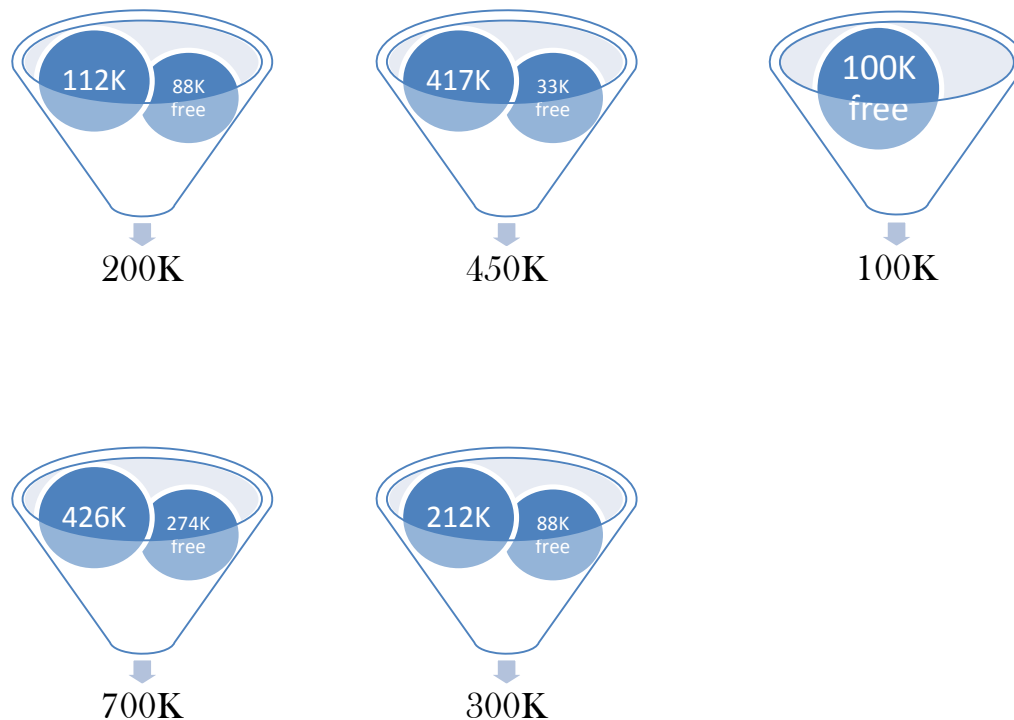


So, with the first fit, there are : $200 + 238 + 100 + 283 + 118 = 939K$ of memory left !!!
 This algorithm is not perfect because we can't fit 426K's process. In order to be placed in a partition, it must wait that the 212K's process is finish (or the 417K).

So, we will suppose that 212K's has finished, now we have:



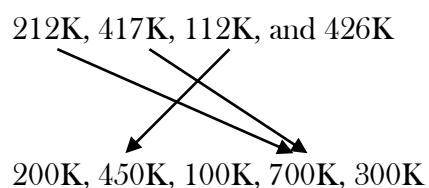
The Best Fit algorithm allocates the smallest hole that is big enough. However, it must search entire list, unless ordered by size and it produces the smallest leftover hole. So, we have the following schema:



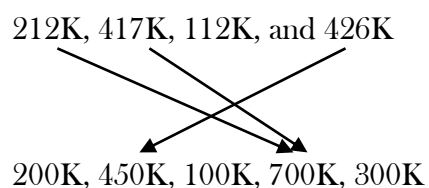
So the Best Fit is a good algorithm because it fit all the processes. Besides there are only $88 + 33 + 100 + 274 + 88 = 583 K$ of memory left.

The Worst fit allocates the largest hole and it must also search the entire list. Besides, it produces the largest leftover hole.

So, this fitting algorithm fills the bigger spaces first, with the original order of processes:



Like the first-fit, the 426K's process cannot be fitted in the memory right now. It has to wait for the 112K process to finish:



So, with the first fit, there are : $200 + (450 - 112) + 100 + (700 - 212 - 417) + 300 = 200 + 338 + 100 + 71 + 300 = 1009K$ of Memory Left.

Conclusion: The best fit algorithm is the best algorithm because it makes the most efficient use of memory.