**Problem-01:**

Consider six memory partitions of size 200 KB, 400 KB, 600 KB, 500 KB, 300 KB and 250 KB. These partitions need to be allocated to four processes of sizes 357 KB, 210 KB, 468 KB and 491 KB in that order.

Perform the allocation of processes using-

1. First Fit Algorithm
2. Best Fit Algorithm
3. Worst Fit Algorithm

**Solution-**

According to question,

The main memory has been divided into fixed size partitions as-



Let us say the given processes are-

* Process P1 = 357 KB
* Process P2 = 210 KB
* Process P3 = 468 KB
* Process P4 = 491 KB

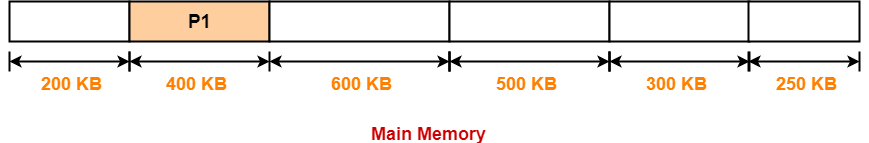
## ****Allocation Using First Fit Algorithm-****

In First Fit Algorithm,

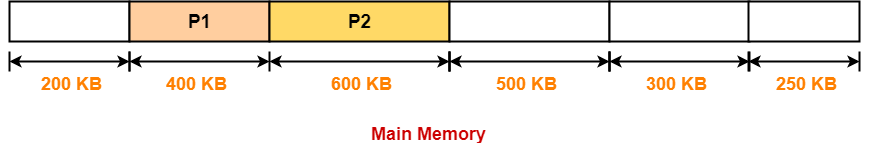
* Algorithm starts scanning the partitions serially.
* When a partition big enough to store the process is found, it allocates that partition to the process.

The allocation of partitions to the given processes is shown below-Ezoic

### **Step-01:**



### **Step-02:**



### **Step-03:**



### **Step-04:**

* Process P4 can not be allocated the memory.
* This is because no partition of size greater than or equal to the size of process P4 is available.

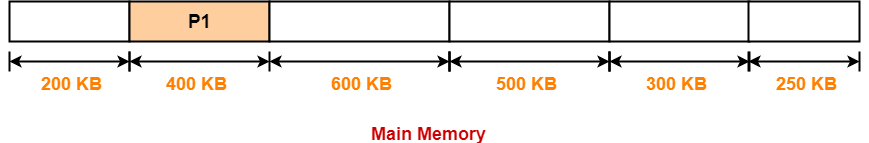
## ****Allocation Using Best Fit Algorithm-****

In Best Fit Algorithm,

* Algorithm first scans all the partitions.
* It then allocates the partition of smallest size that can store the process.

The allocation of partitions to the given processes is shown below-

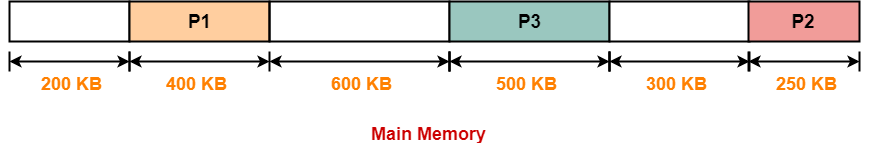
### **Step-01:**



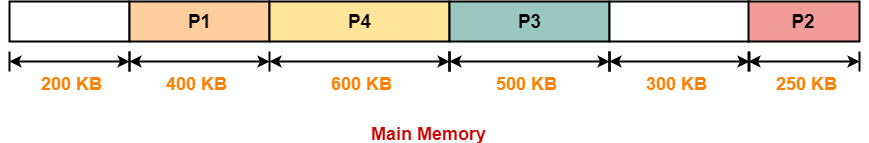
### **Step-02:**



### **Step-03:**



### **Step-04:**



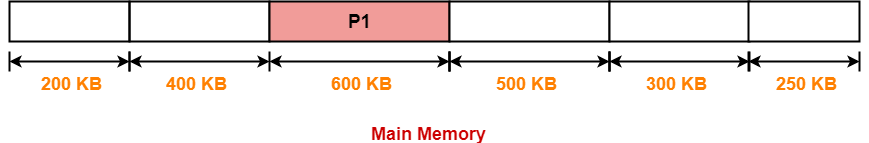
## ****Allocation Using Worst Fit Algorithm-****

In Worst Fit Algorithm,

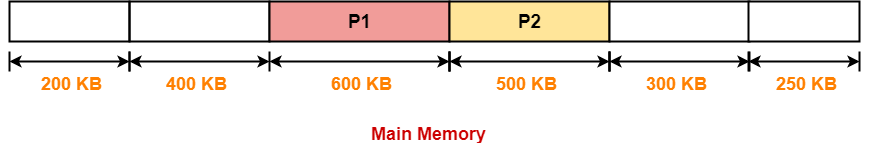
* Algorithm first scans all the partitions.
* It then allocates the partition of largest size to the process.

The allocation of partitions to the given processes is shown below-

### **Step-01:**



### **Step-02:**

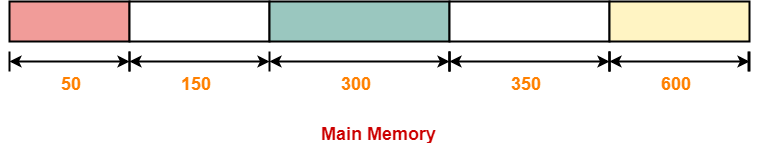


### **Step-03:**

* Process P3 and Process P4 can not be allocated the memory.
* This is because no partition of size greater than or equal to the size of process P3 and process P4 is available.

## ****Problem-02:****

Consider the following heap (figure) in which blank regions are not in use and hatched regions are in use-



Sequence of requests for blocks of size 300, 25, 125, 50 can be satisfied if we use-

1. Either first fit or best fit policy (any one)
2. First fit but not best fit policy
3. Best fit but not first fit policy
4. None of the above

## ****Solution-****

The allocation follows variable size partitioning scheme.

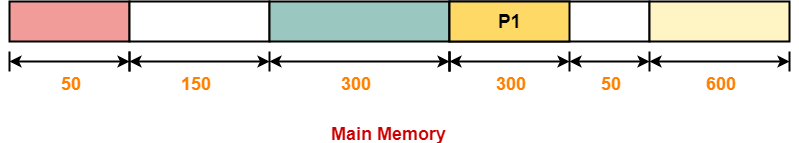
Let us say the given processes are-

* Process P1 = 300 units
* Process P2 = 25 units
* Process P3 = 125 units
* Process P4 = 50 units

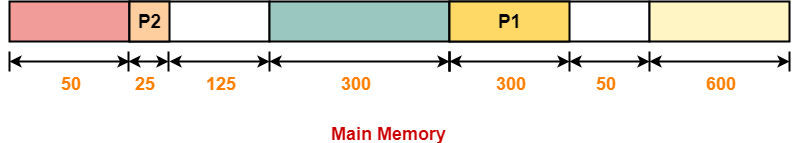
## ****Allocation Using First Fit Algorithm-****

The allocation of partitions to the given processes is shown below-

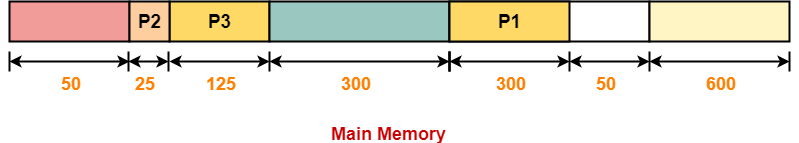
### **Step-01:**



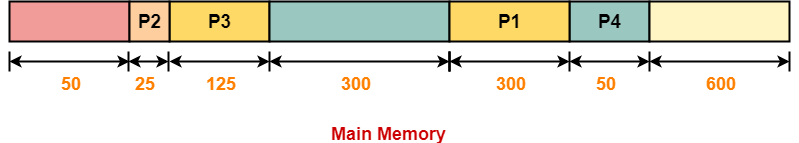
### **Step-02:**



### **Step-03:**



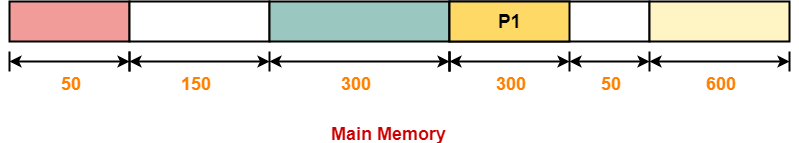
### **Step-04:**



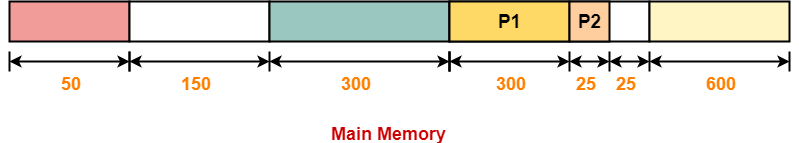
## ****Allocation Using Best Fit Algorithm-****

The allocation of partitions to the given processes is shown below-

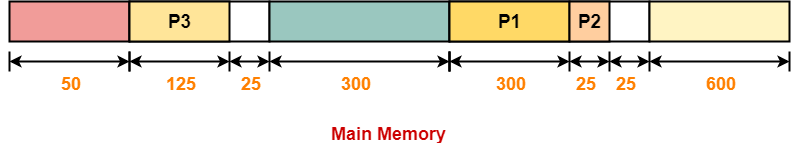
### **Step-01:**



### **Step-02:**



### **Step-03:**



### **Step-04:**

* Process P4 can not be allocated the memory.
* This is because no partition of size greater than or equal to the size of process P4 is available.

Thus,

* Only first fit allocation policy succeeds in allocating memory to all the processes.
* Option (B) is correct.

### **Q. Process requests are given as;**

**25 K , 50 K , 100 K , 75 K**



Determine the algorithm which can optimally satisfy this requirement.

1. First Fit algorithm
2. Best Fit Algorithm
3. Neither of the two
4. Both of them

which can satisfy the request optimally.

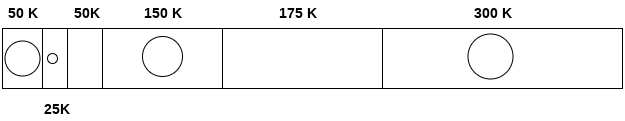
Solution:

## Using First Fit algorithm

Let's see, how first fit algorithm works on this problem.

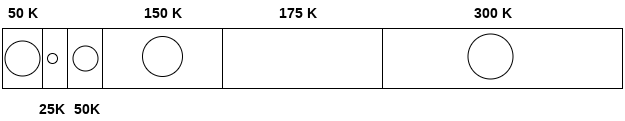
### **1. 25 K requirement**

The algorithm scans the list until it gets first hole which should be big enough to satisfy the request of 25 K. it gets the space in the second partition which is free hence it allocates 25 K out of 75 K to the process and the remaining 50 K is produced as hole.



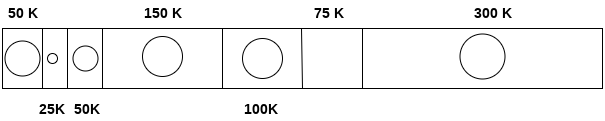
### **2. 50 K requirement**

The 50 K requirement can be fulfilled by allocating the third partition which is 50 K in size to the process. No free space is produced as free space.



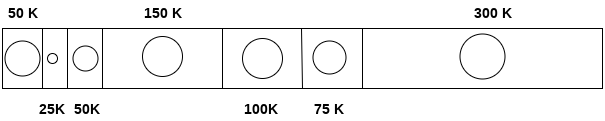
### **3. 100 K requirement**

100 K requirement can be fulfilled by using the fifth partition of 175 K size. Out of 175 K, 100 K will be allocated and remaining 75 K will be there as a hole.



### **4. 75 K requirement**

Since we are having a 75 K free partition hence we can allocate that much space to the process which is demanding just 75 K space.



Using first fit algorithm, we have fulfilled the entire request optimally and no useless space is remaining.

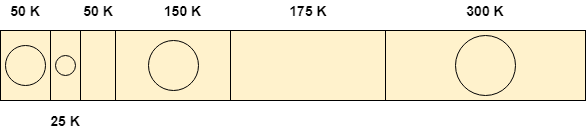
Let's see, How Best Fit algorithm performs for the problem.

## Using Best Fit Algorithm

### **1. 25 K requirement**

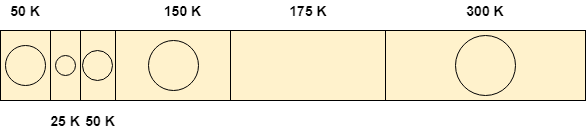
To allocate 25 K space using best fit approach, need to scan the whole list and then we find that a 75 K partition is free and the smallest among all, which can accommodate the need of the process.

Therefore 25 K out of those 75 K free partition is allocated to the process and the remaining 5o K is produced as a hole.



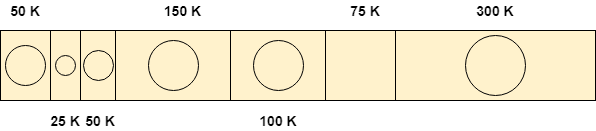
### **2. 50 K requirement**

To satisfy this need, we will again scan the whole list and then find the 50 K space is free which the exact match of the need is. Therefore, it will be allocated for the process.



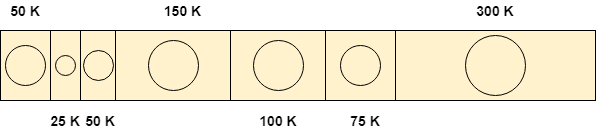
### **3. 100 K requirement**

100 K need is close enough to the 175 K space. The algorithm scans the whole list and then allocates 100 K out of 175 K from the 5th free partition.



### **4. 75 K requirement**

75 K requirement will get the space of 75 K from the 6th free partition but the algorithm will scan the whole list in the process of taking this decision.



By following both of the algorithms, we have noticed that both the algorithms perform similar to most of the extant in this case.

Both can satisfy the need of the processes but however, the best fit algorithm scans the list again and again which takes lot of time.

Therefore, if you ask me that which algorithm performs in more optimal way then it will be **First Fit algorithm** for sure.

Therefore, the answer in this case is A.

**Problem 3:**

Consider five memory partitions of size 100 KB, 500 KB, 200 KB, 450 KB and 600 KB in same order. If sequence of requests for blocks of size 212 KB, 417 KB, 112 KB and 426 KB in same order come, then which of the following algorithm makes the efficient use of memory?

A.Best fit algorithm  
  
B.First fit algorithm  
  
C.Next fit algorithm  
  
D.Both next fit and best fit results in same