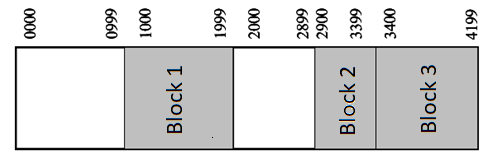
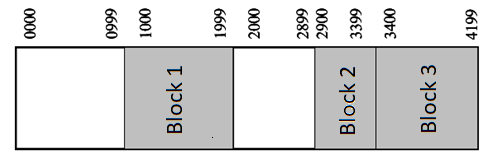
**CONTIGUOUS ALLOCATION (USING BEST-FIT ALGORITHM)**

1. The following exercise will be based on a system with fixed partitioning and a memory with 4200 blocks. The memory has been divided into five partitions of 1000, 1000, 900, 500 and 800 blocks as you can see in the picture below. There are currently three processes.

|  |  |
| --- | --- |
| **Initial Address** | **Length** |
| 1000 | 1000 |
| 2900 | 500 |
| 3400 | 800 |



* 1. From this moment on, we will need to load processes of 500 and 200 data blocks respectively. Draw the memory content after loading these two blocks.



Block 4

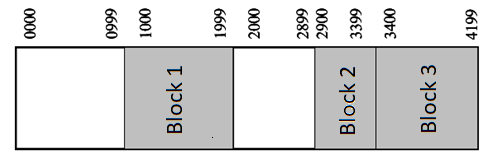
Block 5

* 1. Is there internal fragmentation? Yes

If so, explain where it is and calculate the size. There is an internal fragmentation of 800 blocks in the first partition. There is an internal fragmentation of 400 blocks in the first partition.

* 1. The processes of blocks 2 and 5 have finished. Now, we will try to load a process of 1200 blocks. Is there internal fragmentation? No Explain why or why not Because none of the partitions can load a process of that size.

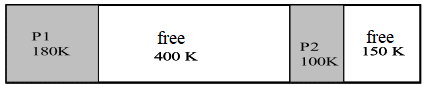
Additional: do the same using first-fit and worst-fit (the answer for both is shown in the picture below)



Block 4

Block 5

1. The following exercise will be based on a system with variable partitioning and a memory. Below, you can see the current state of the memory.



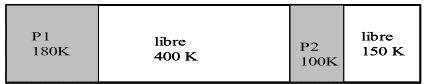
We have the following jobs waiting to be executed: P4 (120K), P5(200K) y P6(90K).

* 1. Suppose none of the current processes finish. Draw the memory content after trying to load these three processes.

How many free space remains? 110K and 30K

Is there internal fragmentation? No

Explain why or why not. We have more partitions to store new processes.



P6

(90K)

P4

(120K)

P5

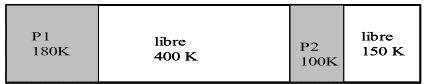
(200K)

* 1. Now, we want to execute P7 (100K). Draw the memory content with the new process.

How many free space remains? 10K and 30K

Is there external fragmentation? Yes

Explain why or why not. There are too many small partitions.



P7

(100K)

P6

(90K)

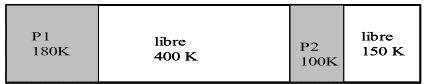
P5

(200K)

P4

(120K)

* 1. Suppose we compact all the free partitions. Draw the memory with all the processes that the system has loaded.



P6

(90K)

FREE

(40K)

P4

(120K)

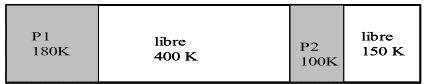
P7

(100K)

P5

(200K)

Additional: do the same using first-fit and worst-fit (the answer for both is shown in the picture below)



P6

(90K)

P5

(200K)

P4

(120K)

After compacting, there will be 140K free blocks that can store P7

1. We have a 512 byte memory managed using paging with 16 byte pages (32 pages in total). At the start, we will have the initial state as shown below.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Process | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 |
| Size (bytes) | 30 | 36 | 8 | 75 | 45 | 60 | 4 | 10 | 25 | 10 |
| Pages | 1,4 | 2,3,10 | 5 | 7,12-15 | 17,19,20 | 23-26 | 22 | 16 | 28,30 | 32 |
| **Fragmentation** | **2** | **12** | **8** | **5** | **3** | **4** | **12** | **6** | **7** | **6** |

Total fragmentation = 65 bytes

1. Draw the state of the memory as it was in its initial stage.
2. Calculate the total fragmentation in the initial stage.

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |
| 16 |  |
| 17 |  |
| 18 |  |
| 19 |  |
| 20 |  |
| 21 |  |
| 22 |  |
| 23 |  |
| 24 |  |
| 25 |  |
| 26 |  |
| 27 |  |
| 28 |  |
| 29 |  |
| 30 |  |
| 31 |  |
| 32 |  |