Memory Management Techniques

Objective

To implement the MFT and MVT Memory Management Techniques.

Description

MFT

MFT (**Multiprogramming with a Fixed number of Tasks**) is one of the old memory management techniques in which the memory is partitioned into fixed size partitions and each job is assigned to a partition. The memory assigned to a partition does not change. It takes place at the time of installation. For example, there can be total 4 partitions and the sizes of each block can be 4KB. Then, the processes which require 4KB or less memory will only get the memory.

At compile time, we can only bind the addresses. This kind of degree of multiprogramming is not flexible because the number of blocks is fixed which cannot be changed which results in memory wastage due to external as well as internal fragmentation.

MVT

MVT (**Multiprogramming with a Variable number of Tasks**) is the memory management technique in which each job gets just the amount of memory it needs. That is, the partitioning of memory is dynamic and changes as jobs enter and leave the system. Hence, there is no partition at the beginning. MVT is a more efficient user of resources. MVT has no internal fragmentation, however, external fragmentation is possible because of which compile time address binding is not possible.

Internal Fragmentation

Whenever a process is loaded or removed from the physical memory block, it creates a small hole in memory space which is called fragment. In case the memory assigned to the process is somewhat larger than the memory requested, then the difference between assigned and requested memory is the internal fragmentation**.**

External Fragmentation

External fragmentation occurs when there is a sufficient amount of space in the memory to satisfy the memory request of a process. But the process's memory request cannot be satisfied as the memory available is in a non-contiguous manner. Either you apply first-fit or best-fit memory allocation strategy it will cause external fragmentation.

Lab Task:

1. Write a program that implements MFT. The program will ask the user the following things.

  i. Total size of available memory in bytes.

ii. Block size in bytes.
iii. Number of processes.
iv. Memory required for each process in bytes.
v. Number of usable blocks available in memory.

Your program will allocate each process with available memory block and will give information about the internal and external fragmentation. Your output should be in tabular form with following entries.

Input

Enter the total memory available (in Bytes) = 1000
Enter the block size (in Bytes) = 300
Enter the number of processes = 5
Enter memory required for process 1 (in Bytes) = 275
Enter memory required for process 2 (in Bytes) = 400
Enter memory required for process 3 (in Bytes) = 290
Enter memory required for process 4 (in Bytes) = 293
Enter memory required for process 5 (in Bytes) = 100
No. of Blocks available in memory = 3

Output

Process No.	Memory Required (B)	Block Allocated (Yes/No)	Internal Fragmentation (B)
1.	275	Yes	25
2.	400	No	-
3.	290	Yes	10
4.	293	Yes	7
5.	100	No	-

Total Internal Fragmentation is 42
Total External Fragmentation is 100

2. Write a program that implements MVT. The program will ask the user the following things.
   i. Total size of available memory in bytes.
   ii. Memory required for each arriving process in bytes.
   iii.

Your program will allocate each process with amount of memory required by the process and will give information about the external fragmentation. Your output should be in tabular form with following entries.

Input

```
Enter the total memory available (in Bytes) = 1000
Enter memory required for process 1 (in Bytes) = 400
Do you want to continue(y/n) = y
Enter memory required for process 2 (in Bytes) = 275
Do you want to continue(y/n) = y
Enter memory required for process 3 (in Bytes) = 550
Do you want to continue(y/n) = n
```

Output

Process No.	Memory Required (B)	Memory Assigned (Yes/No)
1.	400	Yes
2.	275	Yes
3.	550	No

```
Total Memory Allocated is 675
Total External Fragmentation is 325
```