



**King Saud University**  
**College of Computer and Information Sciences Department**  
**Computer Science**  
**CSC227 : Operating Systems**

# **Memory Fragmentation Simulation**

## **Assignment #2**

### **Group #5**

NAME	ID	WORK DIVISON	SECTION#
Shahad Alsuhaibani	444200406	-main and allocatingMemory method and (First-fit) strategy and memoryReport method	52990
Najood Albarrak	444200852	-main and allocatingMemory method and (best-fit) strategy	52990
Noura Alamro	444200941	-main and allocatingMemory method and (worst-fit) strategy	52990
Samar Alghamdi	444201097	-main and deallocatingMemory method and MemoryBlock class	52990

**Supervised by: Dr.Asma Alkhamis**

## Screen shots of sample runs:

### i. Demonstrating sample run included in assignment requirements

```
Enter the total number of blocks: 4
Enter the size of each block in KB: 300 200 100 400
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 2
Memory blocks are created...
Memory blocks:
=====
Block#      size      start-end      status
=====
Block0      300        0-299         free
Block1      200        300-499        free
Block2      100        500-599        free
Block3      400        600-999        free
=====

1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 60
P1 Allocated at address 500, and the internal fragmentation is 40
=====

1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P2 150
P2 Allocated at address 300, and the internal fragmentation is 50
=====
```

```
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 3
Memory blocks:
=====
Block#  size      start-end      status      ProcessID      InternalFragmentation
=====
Block0   300        0-299         free        Null           0
Block1   200        300-499        allocated    P2             50
Block2   100        500-599        allocated    P1             40
Block3   400        600-999        free        Null           0
=====

1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 4
Exiting the program
```

After the user enters number of blocks and their sizes , then chooses the **Best-fit allocation strategy**. An initial blocks report will be printed, After that the user chooses to allocate a memory block and enters the process details, Then the system will allocate the block to the process (**The block that has Free status and the smallest size that fits the process**) and calculate internal fragmentation based on the inputs provided by the user. Finally, when the user chooses the print option from the menu the report will be printed.

## ii. Demonstrating allocation and de-allocation based on Best-fit strategy

```
Enter the total number of blocks: 3
Enter the size of each block in KB: 120 50 80
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 2
Memory blocks are created...
Memory blocks:
=====
Block#    size    start-end    status
=====
Block0    120     0-119       free
Block1    50      120-169     free
Block2    80      170-249     free
=====
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 30
P1 Allocated at address 120, and the internal fragmentation is 20
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P2 100
P2 Allocated at address 0, and the internal fragmentation is 20
=====
```

```

1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 2
Enter the process ID you want to de-allocate: P1
The process P1 is deallocated from the block 1
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 3
Memory blocks:
=====
Block#   size      start-end      status      ProcessID      InternalFragmentation
=====
Block0    120        0-119          allocated    P2              20
Block1    50         120-169        free         Null            0
Block2    80         170-249        free         Null            0
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 4
Exiting the program

Process finished with exit code 0

```

After the user enters number of blocks and their sizes , then chooses the **Best-fit allocation strategy**. An initial blocks report will be printed, After that the user chooses to allocate a memory block and enters the process details, Then the system will allocate the block to the process(**The block that has Free status and the smallest size that fits the process**) and calculate internal fragmentation based on the inputs provided by the user.

**The user will choose to de-allocate a block and enter the process id , then the system will de-allocate the block that contains the process.**

Finally, when the user chooses the print option from the menu the report will be printed.

### iii. Demonstrating the allocation AND de-allocation based on First-fit strategy

```
Enter the total number of blocks: 3
Enter the size of each block in KB: 100 68 200
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 1
Memory blocks are created...
Memory blocks:
=====
Block#    size    start-end    status
=====
Block0    100      0-99        free
Block1    68       100-167     free
Block2    200      168-367     free
=====

1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 20
P1 Allocated at address 0, and the internal fragmentation is 80
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P2 90
P2 Allocated at address 168, and the internal fragmentation is 110
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 2
Enter the process ID you want to de-allocate: P1
The process P1 is deallocated from the block 0
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 3
Memory blocks:
=====
Block#  size    start-end    status    ProcessID    InternalFragmentation
=====
Block0   100      0-99        free      Null         0
Block1   68       100-167     free      Null         0
Block2   200      168-367     allocated  P2           110
=====

1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 4
Exiting the program

Process finished with exit code 0
```

After the user enters number of blocks and their sizes , then chooses the **First-fit allocation strategy**. An initial block report will be printed, After that the user chooses to allocate a memory block and enters the process details, Then the system will allocate the block to the process (**The FIRST block that has Free status and size that fits the process**) and calculates internal fragmentation based on the inputs provided by the user.

**The user will choose to de-allocate a block and enter the process id , then the system will de-allocate the block that contains the process.**

Finally, when the user chooses the print option from the menu the report will be printed.

#### iv. Demonstrating the allocation AND de-allocation based on Worst-fit strategy

```
Enter the total number of blocks: 3
Enter the size of each block in KB: 100 200 300
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 3
Memory blocks are created...
Memory blocks:
=====
Block#      size      start-end      status
=====
Block0      100      0-99          free
Block1      200      100-299       free
Block2      300      300-599       free
=====
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 50
P1 Allocated at address 300, and the internal fragmentation is 250
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 2
Enter the process ID you want to de-allocate: P1
The process P1 is deallocated from the block 2
```

```

1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 200
P1 Allocated at address 300, and the internal fragmentation is 100
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 3
Memory blocks:
=====
Block#   size    start-end    status    ProcessID    InternalFragmentation
=====
Block0    100      0-99        free      Null         0
Block1    200     100-299     free      Null         0
Block2    300     300-599     allocated  P1          100
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 4
Exiting the program

Process finished with exit code 0

```

After the user enters number of blocks and their sizes , then chooses the **Worst-fit allocation strategy**. An initial block report will be printed, After that the user chooses to allocate a memory block and enters the process details, Then the system will allocate the block to the process (**The block that has Free status AND the largest size that fits the process**) and calculates internal fragmentation based on the inputs provided by the user.

**The user will choose to de-allocate a block and enter the process id , then the system will de-allocate the block that contains the process.**

**(If the user re-enters the same process id that has been de-allocated from the block previously, the allocation will be done successfully)**

Finally, when the user chooses the print option from the menu the report will be printed.

#### v. Demonstrating the case when user enters a used process id

```
Enter the total number of blocks: 2
Enter the size of each block in KB: 100 200
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 1
Memory blocks are created...
Memory blocks:
=====
Block#      size      start-end      status
=====
Block0      100        0-99          free
Block1      200       100-299      free
=====
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 50
P1 Allocated at address 0, and the internal fragmentation is 50
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 30
Process id has been used before
Re-enter the process ID and size of process: P2 30
P2 Allocated at address 100, and the internal fragmentation is 170
```

when a user enters a used process id , a message will be printed (Process id has been used before) then, it will ask the user to re-enter different process id and its size.

#### vi. Demonstrating the case when the number of blocks doesn't match the number of entered sizes

```
Enter the total number of blocks: 3
Enter the size of each block in KB: 234 500
Error:The number of blocks (sizes) entered doesn't match number of blocks.
```

An error message will be printed when there is a difference between the number of blocks and the number of entered sizes.



### vii. Demonstrating the case when the user enters an invalid data type

```
Enter the total number of blocks: 2
Enter the size of each block in KB: 700 456
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): g
Please enter a valid data type
```

An error message will be printed that indicates an input mismatch.

### viii. Demonstrating the case when the user chooses an invalid option

```
Enter the total number of blocks: 3
Enter the size of each block in KB: 900 789 400
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 1
Memory blocks are created...
Memory blocks:
=====
Block#    size    start-end    status
=====
Block0    900      0-899       free
Block1    789      900-1688    free
Block2    400      1689-2088   free
=====
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 7
Error:Try again, this choice is invalid.
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 4
Exiting the program

Process finished with exit code 0
```

The system will inform the user that the choice is invalid, and it will ask the user to choose again.

ix. Demonstrating the case when all the memory blocks are allocated to processes

```
Enter the total number of blocks: 2
Enter the size of each block in KB: 100 200
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 2
Memory blocks are created...
Memory blocks:
=====
Block#    size    start-end    status
=====
Block0    100      0-99        free
Block1    200     100-299     free
=====
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 100
P1 Allocated at address 0, and the internal fragmentation is 0
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P2 200
P2 Allocated at address 100, and the internal fragmentation is 0
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P3 400
Error: We can't find an appropriate block to allocate.
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 4
Exiting the program

Process finished with exit code 0
```

When the memory blocks are fully allocated and the user tries to allocate a block for a new process, the system will inform the user that allocation has failed.

x. **Demonstrating the case of de-allocating a process that does not exist**

```
Enter the total number of blocks: 2
Enter the size of each block in KB: 345 678
Enter allocation strategy (1 for first-fit, 2 for best-fit, 3 for worst-fit): 3
Memory blocks are created...
Memory blocks:
=====
Block#    size    start-end    status
=====
Block0    345      0-344        free
Block1    678      345-1022     free
=====
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 1
Enter the process ID and size of process: P1 340
P1 Allocated at address 345, and the internal fragmentation is 338
=====
1) Allocates memory blocks
2) De-allocates memory blocks
3) Print report about the current state of memory and internal Fragmentation
4) Exit
=====
Enter your choice: 2
Enter the process ID you want to de-allocate: P2
Error: We can't find the process P2 in memory.
```

If the user tries to de-allocate a process that isn't in the memory blocks, an error message will be shown.