**COMP 322/L—Introduction to Operating Systems and System Architecture**

**Assignment #4—Memory Allocation**

**Objective:**

To simulate memory allocation with hole-fitting algorithms (First-fit, Best-fit) and implement deallocation and defragmentation of memory blocks.

**Specification:**

The program simulates memory allocation with a chosen hole-fitting algorithm (First-fit, Best-fit) and implements deallocation and defragmentation. A menu controls the operations, and each choice calls the appropriate procedure, where the choices are:

1. Enter parameters
2. Allocate memory for a block
3. Deallocate memory for a block
4. Defragment memory
5. Quit program and free memory

**Assignment:**

* The size of physical memory is represented by an integer *pm\_size*.
* The allocated blocks are contained within a linked list, where each allocated block is a structure containing: (1) the id, (2) the starting address of the block, (3) the ending address of the block, and (4) a link to the next allocated block.
* Each allocation request prompts for: (1) the id and (2) the size of the new block. If the id is a duplicate and/or the remaining physical memory is not enough to fit the request, the request is rejected, and an appropriate message is displayed.
* Each deallocation request prompts for the id. If the id is invalid, the request is rejected.
* Defragmentation compacts the blocks to be contiguous, and coalesces the holes into one hole at the far--right end (highest memory addresses) of physical memory.

**What NOT to do (any violation will result in an automatic score of 0 on the assignment):**

* Do NOT modify the choice values (1,2,3,4,5) or input characters and then try to convert them to integers--the test script used for grading your assignment will not work correctly.
* Do NOT turn in an alternate version of the assignment downloaded from the Internet (coursehero, chegg, reddit, github, etc.) or submitted from you or another student from a previous semester.
* Do NOT turn in your assignment coded in another programming language (C++, C#, Java).

**What to turn in:**

* The source code as a C file uploaded to Canvas by the deadline of 11:59pm PST (-20% per consecutive day for late submissions, up to the 4th day—note 1 minute late counts as a day late, 1 day and 1 minute late counts as 2 days late, etc.)
* Make sure your code compiles with the online C compiler before submitting: <https://www.onlinegdb.com/online_c_compiler>

**Sample output – Best Fit**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 1**

**Enter size of physical memory: 1024**

**Enter hole-fitting algorithm (0=first fit, 1=best\_fit): 1**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 2**

**Enter block id: 0**

**Enter block size: 128**

**ID Start End**

**-------------------**

**0 0 128**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 2**

**Enter block id: 1**

**Enter block size: 320**

**ID Start End**

**-------------------**

**0 0 128**

**1 128 448**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 2**

**Enter block id: 2**

**Enter block size: 224**

**ID Start End**

**-------------------**

**0 0 128**

**1 128 448**

**2 448 672**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 2**

**Enter block id: 3**

**Enter block size: 288**

**ID Start End**

**-------------------**

**0 0 128**

**1 128 448**

**2 448 672**

**3 672 960**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 3**

**Enter block id: 2**

**ID Start End**

**-------------------**

**0 0 128**

**1 128 448**

**3 672 960**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 2**

**Enter block id: 4**

**Enter block size: 128**

**ID Start End**

**-------------------**

**0 0 128**

**1 128 448**

**4 448 576**

**3 672 960**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 3**

**Enter block id: 1**

**ID Start End**

**-------------------**

**0 0 128**

**4 448 576**

**3 672 960**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 2**

**Enter block id: 2**

**Enter block size: 224**

**ID Start End**

**-------------------**

**0 0 128**

**2 128 352**

**4 448 576**

**3 672 960**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 2**

**Enter block id: 5**

**Enter block size: 64**

**ID Start End**

**-------------------**

**0 0 128**

**2 128 352**

**4 448 576**

**3 672 960**

**5 960 1024**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 4**

**ID Start End**

**-------------------**

**0 0 128**

**2 128 352**

**4 352 480**

**3 480 768**

**5 768 832**

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 5**

**Quitting program...**

**Sample output – First Fit**

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 1

Enter size of physical memory: 1024

Enter hole-fitting algorithm (0=first fit, 1=best\_fit): 0

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 2

Enter block id: 0

Enter block size: 128

ID Start End

-------------------

0 0 128

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 2

Enter block id: 1

Enter block size: 320

ID Start End

-------------------

0 0 128

1 128 448

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 2

Enter block id: 2

Enter block size: 224

ID Start End

-------------------

0 0 128

1 128 448

2 448 672

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 2

Enter block id: 3

Enter block size: 288

ID Start End

-------------------

0 0 128

1 128 448

2 448 672

3 672 960

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 3

Enter block id: 2

ID Start End

-------------------

0 0 128

1 128 448

3 672 960

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 2

Enter block id: 4

Enter block size: 128

ID Start End

-------------------

0 0 128

1 128 448

4 448 576

3 672 960

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 3

Enter block id: 1

ID Start End

-------------------

0 0 128

4 448 576

3 672 960

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 2

Enter block id: 2

Enter block size: 224

ID Start End

-------------------

0 0 128

2 128 352

4 448 576

3 672 960

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 2

Enter block id: 5

Enter block size: 64

ID Start End

-------------------

0 0 128

2 128 352

5 352 416

4 448 576

3 672 960

Memory allocation

-----------------

1) Enter parameters

2) Allocate memory for block

3) Deallocate memory for block

4) Defragment memory

5) Quit program

Enter selection: 4

ID Start End

-------------------

0 0 128

2 128 352

5 352 416

4 416 544

3 544 832

**Memory allocation**

**-----------------**

**1) Enter parameters**

**2) Allocate memory for block**

**3) Deallocate memory for block**

**4) Defragment memory**

**5) Quit program**

**Enter selection: 5**

**Quitting program...**