DOCUMENTATION ON THE CODE

Note : for the exemplary purpose I have used a small size in my code which can be increased to the mentioned bigger size later.

1. void initialise\_free(node\* free\_list, int arr[])

Purpose: Initializes the free list with a single node representing the entire available memory.

Arguments:

node\* free\_list: Pointer to the head of the free list

2. void display(node\* list, const string& filename)

Purpose: Writes the contents of a linked list to an output file.

Arguments:

node\* list: Pointer to the head of the linked list to be displayed.

const string& filename: The name of the output file.

3. void update\_allocated(node\* allocated\_list, node\* ptr1)

Purpose: Updates the allocated list by adding a newly allocated node.

Arguments:

node\* allocated\_list: Pointer to the head of the allocated list.

node\* ptr1: Node to be added to the allocated list.

4. void compaction(node\*\* free\_list, node\*\* allocated\_list)

Purpose: Performs memory compaction and coalescing to address fragmentation.

Arguments:

node\*\* free\_list: Pointer to the head of the free list.

node\*\* allocated\_list: Pointer to the head of the allocated list.

5. int allocate(int required\_size, node\*\* free\_list, node\*\* allocated\_list)

Purpose: Allocates a block of memory of the specified size.

Arguments:

int required\_size: Size of the memory block requested.

node\*\* free\_list: Pointer to the head of the free list.

node\*\* allocated\_list: Pointer to the head of the allocated list.

Returns:

Starting address of the allocated block. Returns -1 if allocation fails due to insufficient memory.

6. void deallocate(int add, node\*\* allocated\_list, node\*\* free\_list)

Purpose: Deallocates a specified memory reference if the reference count becomes 0.

Arguments:

int add: Starting address of the memory block to be deallocated.

node\*\* allocated\_list: Pointer to the head of the allocated list.

node\*\* free\_list: Pointer to the head of the free list.

7. int increase\_reference(int a, node\* allocated\_list)

Purpose: Increases the reference count of the specified memory block.

Arguments:

int a: Starting address of the memory block.

node\* allocated\_list: Pointer to the head of the allocated list.

Returns:

The starting address of the memory block.

8. int main()

Purpose: The main function where the program execution starts.

Flow:

Initializes the free list and allocated list.

Reads commands from an input file ("input.txt").

Processes each line, performing allocations, deallocations, and reference count updates based on the commands.

Outputs the final state of the free list and allocated list to separate output files ("free.txt" and "allocated.txt").

INPUTS IN THE INPUT DOCUMENT SHOWN IN DIAGRAM FORM -

