

# OS Assignment 3

## README

2021395, 2022083

MeMS is a simple memory management system implemented in C. It allows for dynamic memory allocation and deallocation using the `mmap` and `munmap` functions. This system maintains a doubly linked list of memory chunks, enabling the splitting and merging of memory as needed.

The **`sys/mman.h`** and **`unistd.h`** libraries are used for memory mapping and other system calls.

### **`mems_init()`**

- The `mems_init` function is responsible for initializing the MeMS system.
- It uses the `mmap` function to allocate memory for the initial node in the linked list.
- If the memory allocation fails, it prints an error message and exits the program.

### **`mems_finish()`**

- The `mems_finish` function is used to deallocate all the memory used by the MeMS system.
- It traverses the linked list and uses the `munmap` function to deallocate memory for each node.
- If any deallocation operation fails, it prints an error message and exits the program.

### **`mems_malloc(size_t size)`**

- The `mems_malloc` function is responsible for allocating memory for the user based on the requested size.
- It traverses the linked list to find a suitable chunk of memory to allocate.
- If no suitable chunk is found, it requests a new chunk from the operating system using `mmap`.
- It also splits the memory chunk if it is larger than the requested size.

### **`mems_free(void *v_ptr)`**

- The `mems_free` function is used to free the memory at the specified virtual address.
- It marks the corresponding node as a hole and attempts to merge adjacent holes if possible to avoid memory fragmentation.

## **mems\_print\_stats()**

- The mems\_print\_stats function prints statistics about the MeMS system.
- It prints information about the main chain length, sub-chain lengths, pages used, and unused space.
- It traverses the linked list and calculates the statistics accordingly.

## **mems\_get(void \*v\_ptr)**

- The mems\_get function is used to find the physical address corresponding to the virtual address provided.
- It traverses the linked list and searches for the node with the specified virtual address.
- If the address is found, it returns the corresponding physical address; otherwise, it prints an error message.

## **main()**

- The main function is used for testing and demonstrating the functionalities of the MeMS system.
- It initializes the MeMS system, allocates and frees memory, assigns values to memory, prints statistics, and finishes the MeMS system.

## **OUTPUT:**

```
kritika395@LAPTOP-87B8MN33:/mnt/c/Kritika/OS_Assign/ass3$ ./example2
----- Allocated virtual addresses [mems_malloc] -----
Virtual address: 140549397196800
Virtual address: 140549397196804
Virtual address: 140549397196808
Virtual address: 140549397196812
Virtual address: 140549397196816
Virtual address: 140549397196820
Virtual address: 140549397196824
Virtual address: 140549397196828
Virtual address: 140549397196832
Virtual address: 140549397196836

This is a test string.

----- Assigning value to Virtual address [mems_get] -----
Virtual address: 0x7fd434e4b000 Physical Address: 0x7fd434e4b000
Value written: 3.140000

----- Printing Stats [mems_print_stats] -----
----- MeMS SYSTEM STATS -----
MAIN[0x7fd434e4b000:0x7fd434e4bfff]-> H[0x7fd434e4b000:0x7fd434e4bfff] <-> NULL
Pages used:
0
Space unused: 4096
Main Chain Length:
1
Sub-chain Length array: [1]

----- Freeing up the memory [mems_free] -----
----- MeMS SYSTEM STATS -----
MAIN[0x7fd434e4b000:0x7fd434e4bfff]-> H[0x7fd434e4b000:0x7fd434e4bfff] <-> NULL
Pages used:
0
Space unused: 4096
Main Chain Length:
1
Sub-chain Length array: [1]

----- MeMS SYSTEM STATS -----
MAIN[0x7fd434e4b000:0x7fd434e4b3e7]-> P[0x7fd434e4b000:0x7fd434e4b3e7] <-> MAIN[0x7fd434e4b3e8:0x7fd434e4bfff]-> H[0x7fd434e4b3e8:0x7fd434e4bfff] <-> NULL
Pages used:
1
Space unused: 3096
Main Chain Length:
2
Sub-chain Length array: [0, 1]

----- Unmapping all memory [mems_finish] -----
```

