OS Lab 1: Memory Management in mCertikOS

Taki Tajwaruzzaman Khan (ID: 210042146)
Tasnim Ashraf (ID: 210042122)

Basic Overview of Tasks

Implemented the physical memory management module for mCertiKOS, divided into three layers:

1. **MATIntro Layer**
2. **MATInit Layer**
3. **MATOp Layer**

MATIntro Layer

Implemented four functions for the Allocation Table (AT):

- `at_is_norm`
- `at_set_perm`
- `at_is_allocated`
- `at_set_allocated`

Solution Approach: MATIntro Layer

- `at_is_norm`:
  - Checks if a page has normal permissions (`perm > 1`).
- `at_set_perm`:
  - Sets the permission of a page.
  - Marks the page as unallocated.
- `at_is_allocated`:
  - Checks if a page is allocated (`allocated > 0`).
- `at_set_allocated`:
  - Sets the allocation status of a page.

MATInit Layer

Function implemented:

- `pmem_init`

Subtasks:

1. Calculate the total number of physical pages (`nps`).
2. Initialize the Allocation Table based on the physical memory map.

Solution Approach: MATInit Layer (1)

Calculate Total Number of Physical Pages

- Retrieved the size of the memory map table using `get_size()`.
- If the table is not empty:
  - Obtained starting address and length of the last memory region.
  - Calculated the highest physical address.
  - Computed `nps` by dividing the highest address by `PAGESIZE`.
- Set `nps` using `set_nps(nps)`.

Solution Approach: MATInit Layer (2)

Initialize the Allocation Table

- Set permission `1` (Kernel only) for pages:
  - From index `0` to `VM_USERLO_PI - 1`
  - From `VM_USERHI_PI` to `nps - 1`
- Initialized user pages with permission `0` (unusable).
- For each entry in the memory map table:
  - Determined if the region is usable (`perm = 2`) or not (`perm = 0`).
  - Calculated the starting page index, adjusted for alignment.
  - Set permissions for pages fully within usable regions.

MATOp Layer

Implemented functions:

- `palloc`
- `pfree`

Solution Approach: MATOp Layer

palloc Function

- Used a static pointer to remember the last checked page index.
- Searched from the pointer for an unallocated, normal permission page.
- Implemented wrap-around to search the entire user space.
- If a page is found:
  - Marked it as allocated using `at_set_allocated`.
  - Updated the pointer.
- Returned `0` if no page is available.

pfree Function

- Marked the page as unallocated using `at_set_allocated(page_index, 0)`.

Thank You!