**Project 2: Pager – A Virtual Memory Manager**  
James Miller  
Grand Canyon University  
CST 315: Operating Systems  
Dr. Citro  
Feb 27, 2023

**GitHub Link**

<https://github.com/jammil002/Virtual-Memory-Manager>

**Project Description**

The provided code demonstrates a simple memory management system using paging, with a virtual memory space consisting of pages, and a physical memory space consisting of frames. The code simulates the process of allocating pages to frames when required and demonstrates how the Least Recently Used (LRU) page replacement algorithm works when the physical memory is full.

**Methodology**

**Definitions and Design**

1. ***The size of physical memory (frame) available (also known as page frame):***

The size of physical memory available is stored in: , which is set to 32. This means there are 32/4=8 frames available, as each frame can hold 4 strings.

1. ***The size of virtual memory available:***

The size of virtual memory available: , which is set to 16. This means there are 16/4=4 virtual pages available.

1. ***Calculate how many pages does each process need, and allocate kernel pages and virtual pages as needed, within defined constraints.***

The code allocates 4 specific logical addresses (2, 7, 12, and 1) to physical memory.

1. ***How will you manage page tables? What data structures will you define? How will you manage page table allocations?***

Page tables are managed using an array named of size which is set to 32. It maps logical pages to the physical frame. The code initializes the page table with -1 values and updates it in the function.

1. ***How will you manage page frames? What data structures will you define? How will you manage frame table allocations?***

Page frames are managed using a 2D string array called . This array represents the physical memory space and is divided into frames. Each row in the array corresponds to a frame, and each frame can hold several strings. The size of physical memory is defined by the constant .

1. ***What memory allocation algorithm will be implemented?***

The memory allocation algorithm is the Demand Paging algorithm combined with the Least Recently Used (LRU) page replacement algorithm. Demand Paging is an approach in which pages are loaded into physical memory only when they are needed. When a page is requested, the code checks if it's already present in physical memory. If not, a page fault is then triggered, and the requested page is loaded into a free frame in physical memory. If no free frames are available, the LRU page replacement algorithm is used to select which frame should be replaced.

1. ***How will you decide when and how to swap pages between the virtual and physical memory?***

The code decides when to swap pages between virtual and physical memory based on the demand paging approach. Pages from virtual memory (logical memory) are loaded into physical memory only when they are requested and not already present in physical memory.

1. ***How will you address and manage page faults and segmentation faults?***

When a logical address is passed to the function, it first checks if the corresponding page is already in physical memory by looking up the . If it's in physical memory checked using the if statement , there is no page fault, and the function proceeds to access the data in the physical memory. If the corresponding page is not in physical memory, a page fault occurs. The code then tries to find a free frame in the physical memory using the function. If there is no free frame available, the least recently used frame replacement algorithm is applied using the function to find a frame to evict. The evicted frame is cleared, and the required page is loaded from logical memory (virtual memory) into the physical memory frame. The is updated accordingly. The access counter is incremented, and the last access time for the frame is updated.

**Code Snippets**

**References**

Gate Smashers [@GateSmashers]. (2018, April 14). *L-5.25: Least Recently Used page replacement algorithm | operating system*. Youtube. <https://www.youtube.com/watch?v=dYIoWkCvd6A>