

Assignment 6

Please complete a report and upload the corresponding codes.

The files should be uploaded directly without compression **without compression without compression without compression**

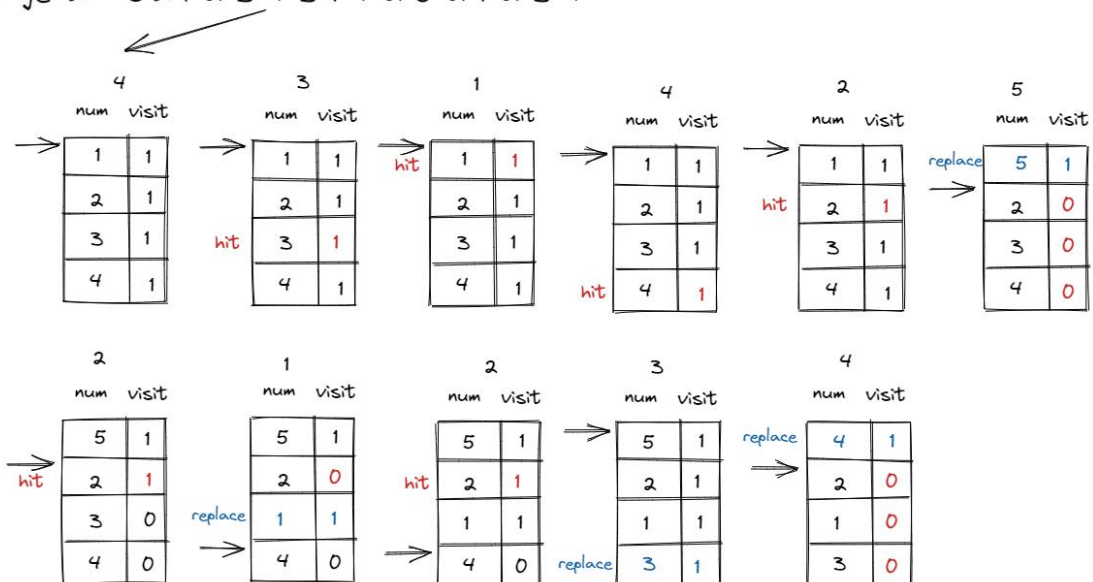
The files to be submitted for this assignment are:

1. report.pdf
2. swap_clock.c

1. **[50pts]** Read Chapter 21 of "Three Easy Pieces" (<https://pages.cs.wisc.edu/~remzi/OSTEP/vm-beyondphys.pdf>) and explain what happens when the process accesses a memory page not present in the physical memory.
2. **[50pts]** Realize Clock algorithm in `swap_clock.c`

Clock: Clock algorithm organizes pages into a circular linked list, just like a clock. The pointer points to the pages loaded the earliest. Besides, Clock algorithm requires a flag bit in PTE to indicate if the corresponding page is used. When the page is used, MMU in CPU will set the flag to 1. When system needs to replace a page, system read the PTE pointer points to, and replace it if the flag is 0, otherwise read the next PTE. The algorithm demonstrates the idea of LRU, and is easy to implement and costs less, but require hardware support to set an reference bit. Clock algorithm is the same as LRU essentially, but it skips pages with reference bit of 1.

page to visit: 1 2 3 4 3 1 4 2 5 2 1 2 3 4



Code you may need:

```
list_entry_t *curr_ptr;  
  
*ptr_page = le2page(le, pra_page_link);  
  
pte_t* ptep = get_pte(mm->pgdir, ptr_page->pra_vaddr, 0);  
  
bool accessed = *ptep & PTE_A;
```

Please realize algorithm in `swap_clock.c`. Change `sm` in `swap.c` to `clock` to test your code.

Please take screen-shot of your code(with annotations) and the running result.