**Lab Test 2 for Section L04**
**Operating Systems SFWRENG 3SH3 Term 2, Winter 2023**
Prof. Neerja Mhaskar

1. Make sure to submit a version of your lab test ahead of time to avoid last minute uploading issues.
2. Note that groups copying each other's solution will get a zero. This includes (but is not limited to) sharing your solution with students taking the test during a different lab hour.
3. Submit your solution under
   Assessments -> Assignments -> Lab Test 2 – Section L04 -> [lab group folder].
4. **Only one copy of your solution should be submitted**.
5. There is one deliverable for this lab test labtest2.c
6. In your c file on the top add your Group number, name(s) and MACID(s) as comments.

**Outline**

Assume that a system has a 32-bit virtual address with a 1024 (=$2^{10}$) byte page size. The physical memory address is also a 32-bit address. Consider a small program that needs only **10** pages of memory.

In today's test you are to write a C program which simulates an MMU's address translation capability. To simulate a program's memory address requests, we use the text file named **ltaddr.txt**. This file can be downloaded from from Assessments -> Assignments -> Lab Test 2 – Section L04. This file contains a sample of **15** logical addresses generated for this test.

1. Your program should read each logical address from this file, compute its corresponding page number (p) and offset (d) using "Bitwise operators in C".

2. The page table is stored in the following binary file **`pagetable.bin`**. This file contains all the frame numbers in binary format. The first frame number stored in the file corresponds to the first page number, the second frame number in the file corresponds to the second page number, and so on. This file can be downloaded from Assessments -> Assignments -> Lab Test 2 – Section L04. You may assume that the frame numbers are **4-byte** integers.

3. You are to memory map the **`pagetable.bin`** file using the `mmap()` system call. After which you will copy each entry into an **integer array** named **`page_table`** using the `memcpy()` function. After copying all entries, you will un-map the file and close it.

4. All subsequent page table access should be done using the `page_table` array.

5. Your program should retrieve the frame number (f) corresponding to the page number (p) from the `page_table` array.

6. After which your program should compute the physical address using the frame number (f) and offset (d) using “Bitwise operators in C”.

**Program Output:** Your program should print each logical/virtual address (read from the file ltaddr.txt), its corresponding page number, page offset and physical address on the console.

Sample Output: `./labtest2`

---

Virtual addr is 567: Page# = 0 & Offset = 567 frame number = 12 Physical addr = 12855.
Virtual addr is 907: Page# = 0 & Offset = 907 frame number = 12 Physical addr = 13195.
Virtual addr is 7401: Page# = 7 & Offset = 233 frame number = 80 Physical addr = 82153.
Virtual addr is 3801: Page# = 3 & Offset = 729 frame number = 122 Physical addr = 125657.
Virtual addr is 4404: Page# = 4 & Offset = 308 frame number = 224 Physical addr = 229684.
Virtual addr is 8399: Page# = 8 & Offset = 207 frame number = 1040 Physical addr = 1065167.
Virtual addr is 2430: Page# = 2 & Offset = 382 frame number = 92 Physical addr = 94590.
Virtual addr is 1480: Page# = 1 & Offset = 456 frame number = 68 Physical addr = 70088.
Virtual addr is 2455: Page# = 2 & Offset = 407 frame number = 92 Physical addr = 94615.
Virtual addr is 9712: Page# = 9 & Offset = 496 frame number = 4050 Physical addr = 4147696.
Virtual addr is 5701: Page# = 5 & Offset = 581 frame number = 40 Physical addr = 41541.
Virtual addr is 3100: Page# = 3 & Offset = 28 frame number = 122 Physical addr = 124956.
Virtual addr is 6144: Page# = 6 & Offset = 0 frame number = 75 Physical addr = 76800.
Virtual addr is 6785: Page# = 6 & Offset = 641 frame number = 75 Physical addr = 77441.
Virtual addr is 7650: Page# = 7 & Offset = 482 frame number = 80 Physical addr = 82402.