بسم الله الرحمن الرحيم

An-Najah National University

Faculty of Engineering and Information Technology

Computer Engineering Department

جامعة النجاح الوطنية

كلية الهندسة وتكنولوجيا المعلومات

قسم هندسة الحاسوب

Operating Systems

Assignment #3 (final): 2nd Semester 2025

Date: 17th/May/2025

Due date: 29th/May/2025

Assignment weight: 5 %

Assignment Objective: This assignment introduces key concepts in memory management by simulating two fundamental mechanisms used in operating systems. In the first part, you will implement virtual-to-physical address translation using a page table and a Translation Lookaside Buffer (TLB), reinforcing your understanding of paging, address decomposition, and caching behaviour. In the second part, you will simulate dynamic contiguous memory allocation, handling memory requests, releases, compaction, and reporting using different allocation strategies such as first-fit and best-fit.

Details: Assume that a system has a 32-bit virtual address space and a page size of X KB. Write a C program that simulates virtual memory translation using a simple page table and a small fully-associative Translation Lookaside Buffer (TLB) with M entries. Total number of frames in the system is assumed to be N. See **Submission Rules** section for more details.

Your program will read a list of **virtual addresses (in decimal)** from a file (provided to you), and for each address, it will output:

- The page number
- The offset
- Whether the TLB resulted in a **hit** or **miss**
- The frame number (from simulated page table)
- The translated **physical address**

Assume the file `addresses.txt` contains:

```
19986

429496

65536

……..
```

You compile and run your program like this:

./vm_simulator addresses.txt

Assuming X = 4 and M = 6, and N = 1024, then your simulator will output something like:

```
Virtual address: 19986

Page number: 4

Offset: 3602

TLB: Miss

Frame number: 10

Physical address: 44562

--------------------------
```

Details

- Example **page size of 4 KB** (i.e., 2^{12}=4096 bytes), so:
  - The **offset** is the lower 12 bits of the virtual address.
  - The **page number** is the upper 20 bits of the virtual address.
- Your simulated **page table** can map page numbers to frame numbers using a simple rule:
  - `frame_number = (page_number + M) % N`
- Implement a small **fully-associative TLB** with **M entries** using **Least Recently Used (LRU)** replacement.
- Your program should maintain a count of **TLB hits and misses**, and print a summary at the end.

Tips

- Use `unsigned int` or `uint32_t` to store 32-bit addresses.
- Use bitwise operations (`>>`, `&`) to extract page number and offset.
- Use arrays and structs to simulate the page table and TLB.

بسم الله الرحمن الرحيم

An-Najah National University

Faculty of Engineering and Information Technology

Computer Engineering Department

جامعة النجاح الوطنية

كلية الهندسة وتكنولوجيا المعلومات

قسم هندسة الحاسوب

Submission Rules:

- All code must be in **c** and must be tested on RHEL variant (RedHat, Centos, …etc)
- Submit working .c file(s) and a **1-2 page PDF report** explaining your work.
- Deadline is 29th /May/2025. Late submissions are allowed for up to 5 days with a fixed penalty of 1 mark (the weight of the assignment becomes 4%).
- You may work individually or in teams of two.
  - If you work in teams of two, **then both team members must submit the code and PDF report on Moodle, clearly mentioning the team members on the first page of the report.**
- Assignment weight is 5% of the course's total mark.

- **For X, M, and N use the least significant digit of your student ID in the following table:**

Least significant digit	X	M	N
0-1	2	4	1024
2-3	4	6	2048
4-5	8	8	4096
6-7	16	10	8192
8-9	32	12	16384

- **If you are working in teams of two, then add both your student IDs, then use the least significant digit of the sum to find X, M and N from the table above.**