# Lab 3: Page Replacement Algorithms

## Lab Objectives

- This lab aims to provide a deep understanding of **page replacement algorithms**, a core component of operating system memory management, by implementing and simulating several classic algorithms.

## Tasks

- Read the code file “page_replacement.c” and understand the implementation of the First In First Out (FIFO) algorithm.
- Complement the code file by implementing:
  1) the Clock algorithm
  2) the Least Recently Used (LRU) algorithm

## Execution

1. Change the current working directory to the location of codes
2. Compile the source code: `gcc page_replacement.c –o test`
3. Run the executable file with input filename as the argument: `./test input.txt`

## Input Format

- The first line has two elements M and N. M represents the maximum size of the page table and N represents the number of pages will be used. The second line is the sequence of page IDs.

## Example Input

```
[yuhanyi@localhost lab3]$ more input.txt
3 10
0 1 2 0 1 3 2 4 3 1
```

# Example Output

```
Max frame number is 3
               ************schedule algorithm************
               1.FIFO
               2.LRU
               3.Clock
 choose the algorithm:
1
Page 0 loaded
Page Table: 0  -  -
Page 1 loaded
Page Table: 0 1  -
Page 2 loaded
Page Table: 0 1 2
Page 0 is already in memory, no page replacement
Page Table: 0 1 2
Page 1 is already in memory, no page replacement
Page Table: 0 1 2
Page 0 replaced by Page 3
Page Table: 3 1 2
Page 2 is already in memory, no page replacement
Page Table: 3 1 2
Page 1 replaced by Page 4
Page Table: 3 4 2
Page 3 is already in memory, no page replacement
Page Table: 3 4 2
Page 2 replaced by Page 1
Page Table: 3 4 1
Total Page Faults: 6
```