KWANGWOON UNIVERSITY

2024년 2학기 **운영체제실습** 11주차

Page Replacement

System Software Laboratory
School of Computer and Information Engineering
Kwangwoon Univ.

Assignment 4-2

- **Simulate four page replacement algorithms:**
  - **Optimal**, **FIFO (First In, First Out)**, **LRU (Least Recently Used)**, and **Clock**.

- **Purpose of the simulation**
  - Observe and compare the performance of each algorithm
  - Number of **page faults** and the **page fault rate**.

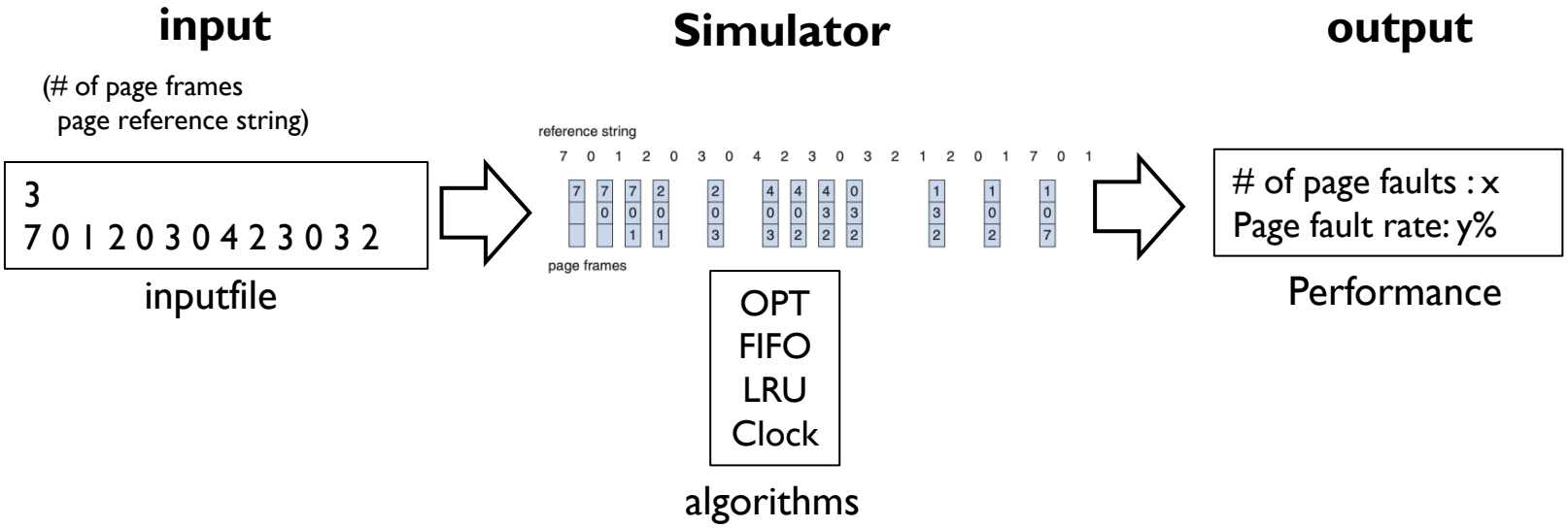
- **Program will read the followings from an input file.**
  - Number of available page frames
  - Page reference strings

Project Description

- **Implement four page replacement algorithms.**
  - **Optimal (OPT)**: Replaces the page that will not be used for the longest period in the future.
  - **FIFO (First In, First Out)**: Replaces the **oldest page** that was loaded into memory.
  - **LRU (Least Recently Used)**: Replaces the page that has not been used for the longest period.
  - **Clock Algorithm**: Circular structure with a reference bit to determine which pages to replace, giving pages a "second chance" before replacing them.

Project Description

- **Page replacement simulator overview**



Project requirements

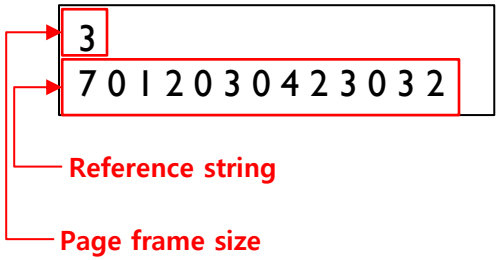
- **1. Implement four page replacement algorithms:**
  - **Optimal**, **FIFO**, **LRU**, and **Clock** Algorithm.
- **2. Input Handling**
  - Read the number of frames and the page reference string from an input file.
- **3. Simulate the page replacement process and track the number of page faults.**
  - Page frames are assumed to be initially empty.
- **4. Print the number of page faults and page fault rate for each algorithm.**

- **Note: If you use a static array to implement the number of page frames, assume the maximum is 1,000.**

Input

- **Page reference string**
  - Read from an input file.
  - Input file contains two parts:
    - **Number of page frames**: An integer representing the number of available frames.
    - **Page reference string**: Integers represent the pages requested by the system in order.

  - Input file example (input.1)



Simulator

- **Command-line Usage**
  - **“page_replacement_simulator inputfile”**
    - **Read the number of frames and page reference string from the inputfile**
    - **Simulate the four algorithms (Optimal, FIFO, LRU, and Clock).**
  - Examples
    - **Page_replacement_simulator input.1**
      - Simulate all page replacement algorithms using the data file "input.1".

Output

- **Sample output**

```
Optimal Algorithm:
Number of Page Faults: X
Page Fault Rate: Y%

FIFO Algorithm:
Number of Page Faults: X
Page Fault Rate: Y%

LRU Algorithm:
Number of Page Faults: X
Page Fault Rate: Y%

Clock Algorithm:
Number of Page Faults: X
Page Fault Rate: Y%
```

- **보고서에 각 알고리즘 별 성능결과를 분석 비교할 것.**

Test Case 01.

- **Input file(input.1)**

```
os2024123456@ubuntu:~/assgin4/4-2$ cat input.1
3
7 0 1 2 0 3 0 4 2 3 0 3 2
```

- **$./page_replacement_simulator input.1**

```
os2024123456@ubuntu:~/assgin4/4-2$ ./page_replacement_simulator input.1
Optimal Algorithm:
Number of Page Faults: 7
Page Fault Rate: 53.85%

FIFO Algorithm:
Number of Page Faults: 10
Page Fault Rate: 76.92%

LRU Algorithm:
Number of Page Faults: 9
Page Fault Rate: 69.23%

Clock Algorithm:
Number of Page Faults: 8
Page Fault Rate: 61.54%
```

Test Case 01.

- **Optimal Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13
	7	0	1	2	0	3	0	4	2	3	0	3	2
-	7	7	7	2	2	2	2	2	2	2	2	2	2
-		0	0	0	0	0	0	4	4	4	0	0	0
-			1	1	1	3	3	3	3	3	3	3	3
	F	F	F	F	H	F	H	F	H	H	F	H	H

Page Frame

Page Fault

- **FIFO Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13
	7	0	1	2	0	3	0	4	2	3	0	3	2
-	7	7	7	2	2	2	2	4	4	4	0	0	0
-		0	0	0	0	3	3	3	2	2	2	2	2
-			1	1	1	1	0	0	0	3	3	3	3
	F	F	F	F	H	F	F	F	F	F	F	H	H

KWANGWOON UNIVERSITY

Test Case 01.

- **LRU Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13
	7	0	1	2	0	3	0	4	2	3	0	3	2
-	7	7	7	2	2	2	4	4	4	4	0	0	0
-		0	0	0	0	0	0	0	0	3	3	3	3
-			1	1	1	3	3	2	2	2	2	2	2
	F	F	F	F	H	F	H	F	F	F	F	H	H

- **clock Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13
	7	0	1	2	0	3	0	4	2	3	0	3	2
-	7(0)	7(0)	7(0)	2(0)	2(0)	2(0)	2(0)	4(0)	4(0)	3(0)	3(0)	3(1)	3(1)
-		0(0)	0(0)	0(0)	0(1)	0(0)	0(1)	0(1)	0(0)	0(0)	0(1)	0(1)	0(1)
-			1(0)	1(0)	1(0)	3(0)	3(0)	3(0)	2(0)	2(0)	2(0)	2(0)	2(1)
	F	F	F	F	H	F	H	F	F	F	H	H	H

Test Case 02.

- **Input file(input.2)**

```
os2024123456@ubuntu:~/assgin4/4-2$ cat input.2
3
1 2 3 4 2 1 2 1 3 1
```

- **$./page_replacement_simulator input.2**

```
os2024123456@ubuntu:~/assgin4/4-2$ ./page_replacement_simulator input.2
Optimal Algorithm:
Number of Page Faults: 5
Page Fault Rate: 50.00%

FIFO Algorithm:
Number of Page Faults: 7
Page Fault Rate: 70.00%

LRU Algorithm:
Number of Page Faults: 6
Page Fault Rate: 60.00%

Clock Algorithm:
Number of Page Faults: 6
Page Fault Rate: 60.00%
```

Test Case 02.

- **Optimal Algorithm**

	1	2	3	4	5	6	7	8	9	10
	1	2	3	4	2	1	2	1	3	1
-	1	1	1	1	1	1	1	1	1	1
-		2	2	2	2	2	2	2	2	2
-			3	4	4	4	4	4	3	3
	F	F	F	F	H	H	H	H	F	H

Page Frame

Page Fault

- **FIFO Algorithm**

	1	2	3	4	5	6	7	8	9	10
	1	2	3	4	2	1	2	1	3	1
-	1	1	1	4	4	4	4	4	3	3
-		2	2	2	2	1	1	1	1	1
-			3	3	3	3	2	2	2	2
	F	F	F	F	H	F	F	H	F	H

KWANGWOON UNIVERSITY

Test Case 02.

- **LRU Algorithm**

	1	2	3	4	5	6	7	8	9	10
	1	2	3	4	2	1	2	1	3	1
-	1	1	1	4	4	4	4	4	3	3
-		2	2	2	2	2	2	2	2	2
-			3	3	3	1	1	1	1	1
	F	F	F	F	H	F	H	H	F	H

- **clock Algorithm**

	1	2	3	4	5	6	7	8	9	10
	1	2	3	4	2	1	2	1	3	1
-	1(0)	1(0)	1(0)	4(0)	4(0)	4(0)	4(0)	4(0)	3(0)	3(0)
-		2(0)	2(0)	2(0)	2(1)	2(0)	2(1)	2(1)	2(1)	2(1)
-			3(0)	3(0)	3(0)	1(0)	1(0)	1(1)	1(1)	1(1)
	F	F	F	F	H	F	H	H	F	H

Test Case 03.

- **Input file(input.3)**

```
os2024123456@ubuntu:~/assgin4/4-2$ cat input.3
4
1 3 0 3 5 6 3 2 5 2 4 1 0 5
```

- **$./page_replacement_simulator input.3**

```
os2024123456@ubuntu:~/assgin4/4-2$ ./page_replacement_simulator input.3
Optimal Algorithm:
Number of Page Faults: 8
Page Fault Rate: 57.14%

FIFO Algorithm:
Number of Page Faults: 10
Page Fault Rate: 71.43%

LRU Algorithm:
Number of Page Faults: 10
Page Fault Rate: 71.43%

Clock Algorithm:
Number of Page Faults: 9
Page Fault Rate: 64.29%
```

Test Case 03.

- **Optimal Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	1	3	0	3	5	6	3	2	5	2	4	1	0	5
Page Frame -	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Page Frame -		3	3	3	3	3	3	3	3	3	3	3	3	3
Page Frame -			0	0	0	6	6	2	2	2	4	4	0	0
Page Frame -					5	5	5	5	5	5	5	5	5	5
Page Fault	F	F	F	H	F	F	H	F	H	H	F	H	F	H

- **FIFO Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	1	3	0	3	5	6	3	2	5	2	4	1	0	5
Page Frame -	1	1	1	1	1	6	6	6	6	6	6	6	0	0
Page Frame -		3	3	3	3	3	3	2	2	2	2	2	2	5
Page Frame -			0	0	0	0	0	0	0	0	4	4	4	4
Page Frame -					5	5	5	5	5	5	5	1	1	1
Page Fault	F	F	F	H	F	F	H	F	H	H	F	F	F	F

Test Case 03.

- **LRU Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	1	3	0	3	5	6	3	2	5	2	4	1	0	5
-	1	1	1	1	1	6	6	6	6	6	4	4	4	4
-		3	3	3	3	3	3	3	3	3	3	1	1	1
-			0	0	0	0	0	2	2	2	2	2	2	5
-					5	5	5	5	5	5	5	5	0	0
	F	F	F	H	F	F	H	F	H	H	F	F	F	F

- **clock Algorithm**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	1	3	0	3	5	6	3	2	5	2	4	1	0	5
-	1(0)	1(0)	1(0)	1(0)	1(0)	6(0)	6(0)	6(0)	6(0)	6(0)	4(0)	4(0)	4(0)	5(0)
-		3(0)	3(0)	3(1)	3(0)	3(0)	3(1)	3(0)	3(0)	3(0)	3(0)	1(0)	1(0)	1(0)
-			0(0)	0(0)	0(0)	0(0)	0(0)	2(0)	2(0)	2(1)	2(1)	2(1)	2(0)	2(0)
-					5(0)	5(0)	5(0)	5(0)	5(1)	5(1)	5(0)	5(0)	0(0)	0(0)
	F	F	F	H	F	F	H	F	H	H	F	F	F	H

Clock algorithm – requirement1

- **1)Initial 후 처음으로 page frame에 참조 시 Page fault 발생**

evict

	1	2	3	
-	1(0)	1(0)	1(0)	
-	-	2(0)	2(0)	…
-	-	-	3(0)	
	F	F	F	

Clock algorithm – requirement2

- 2) 본 과제에서 구현 하는 Clock 알고리즘은 다음과 같은 방식을 따릅니다.