

## Contents

<b>A. Introduction</b>	4
<b>Project goal</b>	4
<b>Problem statement</b>	4
<b>Problem Define</b>	4
<b>Problem Solving</b>	4
<b>B. Body</b>	7
<b>개발 환경</b>	7
<b>Implementation</b>	7
<b>Result</b>	11
Error test 및 예외 처리	11
input file을 이용한 test	16
random input	22
<b>C. Conclusion</b>	31
<b>D. Reference</b>	31

## Figure Contents

[1]Figure 1 input file format(Figure Caption) .....	4
Figure 2 use define for flexible range(Figure Caption) .....	7
Figure 3 함수 간 data 이동을 표현한 diagram.....	8
Figure 4 array값 설명.....	9
Figure 5 LFU tie break.....	11
Figure 6 파일이 존재하지 않음 .....	11
Figure 7 page 범위 벗어남 .....	11
Figure 8 page 범위 벗어남 에러 문구.....	11
Figure 9 page 범위 벗어남(마이너스 범위).....	12
Figure 10 page 범위 벗어남(마이너스 범위) 에러 문구 .....	12
Figure 11 reference string 짧음 .....	12
Figure 12 reference string 짧음 에러 문구 .....	12
Figure 13 reference string 김(len: 15).....	12
Figure 14 명시된 것보다 긴 부분은 무시하고 진행 .....	13
Figure 15 첫 줄 정수 개수 작음(3개).....	13
Figure 16 줄에 상관없이 처음 등장하는 정수 4개 사용 .....	14
Figure 17 첫 줄 정수 개수 많음 (5개).....	14
Figure 18 첫 줄 정수 개수 5개 - 4개를 제외한 값은 reference string의 일부로 취급 .....	15
Figure 19 input1.txt console 결과.....	17
Figure 20 input1.txt output file.....	21
Figure 21 frame 3일 때 .....	21
Figure 22 frame 4일 때 .....	22
Figure 23 rand_input.txt.....	23
Figure 24 rand_input.txt의 결과 .....	23
Figure 25 조정된 range.....	24
Figure 26 page 개수 < page frame 개수 input(input6.txt에 저장) .....	24
Figure 27 page 개수 < page frame 개수 output.....	27
Figure 28 WS 취약점(input7.txt에 저장) .....	27
Figure 29 일반적인 random input(input8.txt에 저장).....	28
Figure 31 page >> frame input(input9.txt에 저장).....	29
Figure 32 비교할 random input(input5.txt에 저장).....	29
Figure 33 window size를 10으로 증가(input3.txt).....	30

Figure 34 window size 2로 감소(input4.txt) .....	30
---	----

## Table Contents

Table 1 Figure27 결과 .....	29
Table 2 page >> frame output .....	29
Table 3 비교할 random output .....	30
Table 4 window size 10일 때 .....	30
Table 5 window size 2일 때 .....	30

## A. Introduction

### Project goal

MIN, FIFO, LFU, LRU, Clock, Working Set(WS) 가상 메모리 관리 기법의 replace rule에 따라 page fault 시 victim을 선정하고 총 page fault의 횟수, residence set을 구할 수 있다.

### Problem statement

#### Problem Define

1. MIN, FIFO, LFU, LRU, Clock, Working Set(WS) algorithm에 따라 가상 메모리를 관리한다.
2. input file의 형식은 다음과 같다.

#### ■ 입력 포맷 및 입력 예

N	M	W	K
$r_1$	$r_2$	$r_3$	$r_4 \ r_5 \ \cdots \ r_K$

6	3	3	15
0	1	2	3 2 3 4 5 4 1 3 4 3 4 5

- N은 process가 갖는 page 개수 (최대 100)
  - Page 번호는 0번부터 시작
- M은 할당 page frame 개수 (최대 20, WS 기법에서는 비사용)
- W는 window size (최대 100, WS 기법에서만 사용)
- K는 page reference string 길이 (최대 100,000)
- " $r_1 \ r_2 \ r_3 \ r_4 \ r_5 \ \cdots \ r_K$ "는 page reference string

[1]Figure 1 input file format(Figure Caption)

3. simulation을 진행한 후 page fault가 발생한 지점, victim, 발생한 page fault 횟수와 매시간 residence set을 결과로 출력한다.
4. process의 page 개수, 할당 받은 page frame 개수, window size, page reference string은 각각 제한이 있다.
5. 초기 page frame은 모두 비어 있는 것으로 가정한다.

#### Problem Solving

1. 기법마다 replace에 필요한 data가 다르므로 각각의 기법을 수행하면서 이를 저장한다.
  - a) MIN  
forward distance가 가장 큰 page가 교체된다. distance는 page fault

가 발생하였을 때에만 계산한다. 계산한 값 중 가장 큰 값을 가지는 page가 교체된다.

따라서 distance를 저장할 page frame 개수만큼의 크기를 갖는 array가 필요하다.

b) FIFO

arrival time이 가장 작은 page가 교체된다. 새로운 page의 등장은 page fault가 발생할 때만 발생하므로 page fault가 발생하였을 때 새로 등장한 page의 arrival time을 저장할 page frame 개수만큼의 크기를 갖는 array가 필요하다. 이 array에서 가장 작은 값을 갖는 page를 교체한다.

c) LRU

가장 최근에 사용된 시간(used time)이 가장 작은 page가 교체된다. page fault가 일어나지 않아도 해당 page가 memory 내에 있다면 사용될 수 있기 때문에 매시간 사용된 page의 사용된 시간을 저장할 page frame 개수만큼의 크기를 갖는 array가 필요하다. page fault가 발생하면 이 array에서 가장 작은 값을 갖는 page를 교체한다.

d) LFU

가장 사용한 횟수(used count)가 작은 page가 교체된다. page fault가 일어나지 않아도 해당 page가 memory 내에 있다면 사용될 수 있기 때문에 매시간 사용된 page의 사용된 횟수를 저장할 page frame 개수만큼의 크기를 갖는 array가 필요하다. 이 array에서 가장 작은 값을 갖는 page를 교체한다. 만약 같은 값을 갖는 page가 여러 개라면 그 중에서 LRU기법을 이용하여 선택한다. 이를 위해 used time 또한 저장해야 되기 때문에 used count를 저장한 array와 별도로 page frame 개수만큼의 크기를 갖는 array가 필요하다.

e) Clock

reference bit가 0인 page를 교체한다. page fault가 일어나지 않아도 해당 page가 사용되었다면 reference bit을 1로 바꿔야 하므로 page frame 개수만큼의 크기를 갖는 array가 필요하다. clock의 needle은 교체한 page frame의 다음 frame을 가리켜야 한다. 만약 현재 가리키는 frame의 reference bit이 1이라면 이를 0으로 바꾸고 다음으로 넘어간다. needle은 0 ~ no. of page frame - 1 사이의 수를 순차적으로 회전한다. (no. of page frame - 1 다음 frame은 0이다)

f) WS

Working Set은 다른 방법과 달리 variable allocation이므로 page frame의 개수를 고정할 수 없다. 따라서 page의 개수만큼의 크기를 갖는 array를 사용한다. window size보다 이전에 사용된 page는 memory 안에 있을 수 없다. 이를 계산하기 위해서는 page가 쓰인 시간(used time)을 array에 저장해야 한다. 그리고 매 순간 쓰인 page의 used time을 저장한 후 window size보다 이전에 사용된 page는 memory에서 내보내고 그 순간 page fault가 발생한 page는 memory에 load한다. 이 때 load하고 쓰이기 때문에 array에 이 시간을 used time으로 기록해야 한다.

2. input 파일의 종류를 다양하게 하기 위해서 프로그램에서 random input을 생성하는 기능을 추가한다.
  - a) keyboard input: random  
만약 random을 입력하면 프로그램 내에서 자체적으로 random input을 생성하고 이를 파일로 만들어 저장한다.
  - b) keyboard input: 파일 이름  
파일 이름을 입력하면 입력 파일에 저장된 값으로 시뮬레이션을 진행한다.
3. input 파일의 format은 Figure 1을 따른다.
4. 입력 파일에서 첫번째 줄을 읽어오면 정해진 범위를 따르는 지 확인한다. 만약 따르지 않는다면 에러 문구를 출력하며 프로그램을 종료한다.
5. 두번째 줄에서 읽은 값이 첫번째 줄에서 명시된 length of string과 다르다면 에러 문구를 출력하며 프로그램을 종료한다. 그렇지 않다면 읽은 값을 int array에 저장한다.
6. memory 안에 존재하는지 찾는 함수를 통해서 만약 string에서 읽은 page가 memory에 존재하지 않는다면 page fault이고 victim을 선정하여 page를 교체한다. 만약 존재한다면 1.에서 명시한 것처럼 표시할 것이 있다면 표시하고 없다면 string의 page에 대해 검사를 진행한다.

## B. Body

### 개발 환경

1. 사용 언어: C
2. OS: Ubuntu 18.04 (64bit)
3. IDE: code block 17.12
4. compiler: gcc 7.3.0, 32-bit
5. encoding: UTF-8
6. 실행방법: OS2020-2\_2018312292\_김동원\_P3.c를 gcc로 컴파일한 후 실행한다.

### Implementation

A.Problem Solving에서 명시한대로 구현한다.

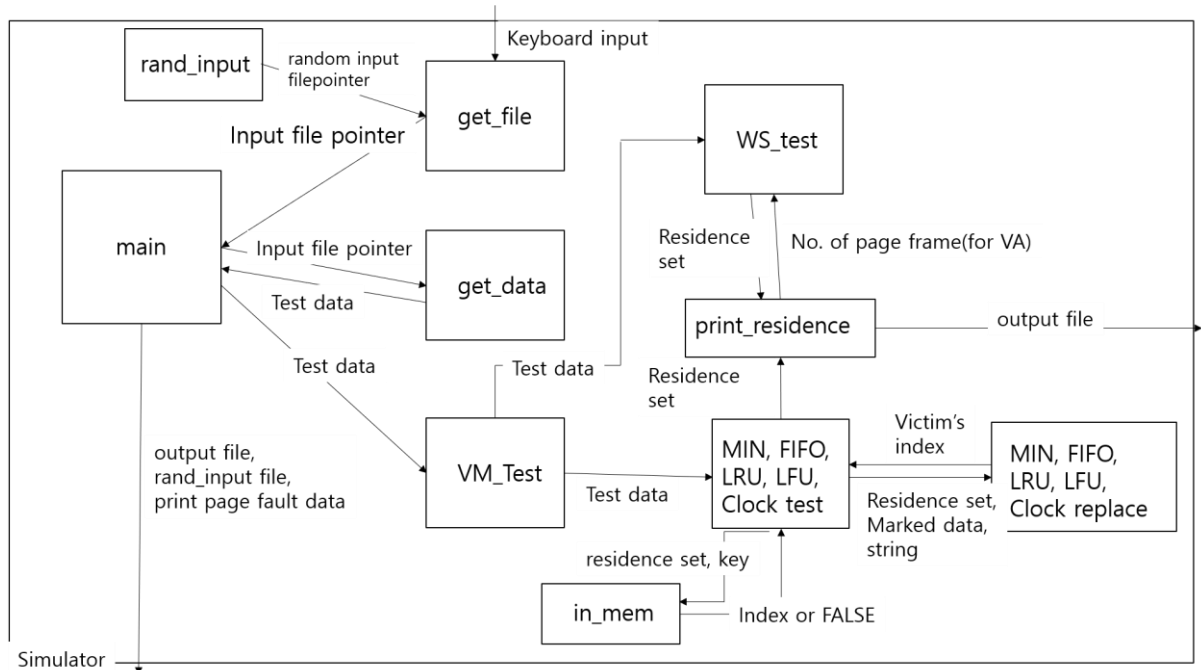
1. 변수의 범위는 define을 이용하여 필요에 따라 바꿀 수 있도록 구현했다.

```
#define FILE_LENGTH 100 // keyboard input length limit
#define PAGE 100 // limit of no. of page
#define FRAME 20 // limit of no. of page frame allocated
#define WINDOW 100 // limit of window size
#define S_LEN 100000 // limit of length of reference string
#define FALSE -1
```

**Figure 2 use define for flexible range(Figure Caption)**

get\_data에서 명시된 범위와 부합하지 않는 입력이 주어지는지 확인하고 만약 그렇다면 에러 문구를 출력하고 -2를 return하며 프로그램을 종료한다. 만약 명시된 string\_length보다 실제 reference string의 길이가 길다면 string\_length만큼만 input으로 받는다. (비록 format에 맞지는 않지만 test를 진행하기 위한 data가 충분하다고 판단하였기 때문에 나머지는 무시한다.)

2. 입력 format은 가독성을 고려하여 Figure1을 따르나 실제 구현에서는 format과 상관없이 파일 내의 첫 4개의 정수를 각각 page 개수, page frame 개수, window 사이즈, reference string의 길이로 받고 시뮬레이션을 진행한다.
3. 함수 간 데이터 이동을 표현한 diagram은 다음과 같다.



**Figure 3 함수 간 data 이동을 표현한 diagram**

- a) input file pointer: FILE \*infile
- b) keyboard input: 동적 메모리 할당을 받은 char \*filename에 저장. 사이즈는 FILE\_LENGTH로 define 되어 있고 필요에 따라 바꿀 수 있음.
- c) random input file pointer: 만약 keyboard input이 random이면 정의된 범위 내의 난수를 생성하여 "rand\_input.txt"를 'w+' mode로 연다. 파일이 존재해도 새로운 내용으로 덮어쓰고 파일이 존재하지 않는다면 새로 만들어야 하는 상황이기 때문에 'w+' mode를 사용하였다. return 된 file pointer는 get\_file 함수를 거쳐 main으로 전달되게 된다.
- d) Test data
  - i. int page: 프로세스의 총 페이지 개수
  - ii. int page\_frame: 프로세스가 할당 받은 페이지 개수. VA인 WS에서는 사용하지 않는다.
  - iii. int window: WS에서 사용될 window size. 다른 기법에서는 사용되지 않는다.
  - iv. int \*string: reference string을 저장한 array. array의 index는 시뮬레이터에서 time으로 취급된다. (시간은 1부터 시작하므로 index + 1 = time이다)
  - v. int string\_length: reference string의 길이



- vi. FILE \*outfile: residence set, page fault가 발생한 시간과 victim, 총 page fault 발생 횟수, 평균 page frame 사용 개수(WS만)을 담은 output file의 포인터이다.
- vii. int \*mem: VS\_Test에서 정의된 동적할당 array로 page frame에 들어있는 page의 번호를 저장한다. FA에 사용된다. (Test에서 replace로 전달되는 Test data에 포함된다.) (크기: page\_frame)
- viii. int \*mark: VS\_Test에서 정의된 동적할당 array로 replace에 필요한 정보(arrival time, forward distance, used time, used count, reference bit)를 저장하는데 사용된다. FA에 사용된다. (Test에서 replace로 전달되는 Test data에 포함된다.) (크기: page\_frame)
- ix. int \*page\_mark: VS\_Test에서 정의된 동적할당 array로 VA에서 frame에 들어있는 페이지에 대해 used time을 저장하는데 사용된다. (VS\_Test에서 WS\_test로 전달되는 Test data에 포함된다.) (크기: page)

mem	[0]	[1]	...	...	...	...	...	...	...	[page_frame - 1]
	page no.	-1	...							

mark	[0]	[1]	...	...	...	...	...	...	...	[page_frame - 1]
	Data of mem[0]	Data of mem[1]	...							

page_mark	[0]	[1]	...	...	...	...	...	...	...	[page - 1]
	used time of page0	used time of page1	...							

**Figure 4 array값 설명**

사용된 array가 저장하고 있는 값은 다음과 같다.

- 1) 모든 array는 -1로 초기화된다.
- 2) mem의 경우 해당 frame이 비어 있다면 -1을 갖는다.
- 3) MIN 기법에서 만약 해당 시간 기준 앞으로 그 page가 사용되지 않는다면 무한대를 넣는 것이 맞으나 이 시뮬레이션에서는 string\_length 값을 저장하였다. 실제로 forward distance는 최대 string\_length - 1까지 가질 수 있기 때문에 무한대를 표현하는 것과 비슷한 효과라고 판단하였기 때문

이다.

- 4) page\_mark의 경우 page가 page frame내에 없을 때 -1 값을 갖는다.
  - e) residence set: FA의 경우 mem에서 -1이 아닌 값(page no.)과 같다. VA의 경우 page\_mark에서 값이 -1이 아닌 index와 같다.
  - f) key: 현재 시간의 string 값 (사용하려는 page no.)  
mem에서 key 값은 linear search로 찾는다. 정렬되어 있지 않은 값 일 뿐 아니라 mem의 크기도 작기 때문이다. 만약 key값이 mem 안에 존재하면 존재하는 frame의 index 값, 존재하지 않는다면 FALSE(-1)을 in\_mem에서 return한다.
  - g) marked data: mark를 의미한다.
  - h) string: Test\_data에서 설명한 reference string과 같은 array이다.
  - i) victim's index: replace에서 mark를 이용하여 고른 교체될 page의 mem 내에서의 page frame 번호이다.
  - j) No. of page frame: VA에서는 단순히 fault ratio가 작은 것뿐 아니라 적은 page frame을 사용하면서 ratio를 줄이는 것이 중요하다. 따라서 WS의 경우 print\_residence에서 해당 시간에 사용된 page frame의 개수를 return한다. WS\_test에서 return 받은 값을 모두 더한 후 string\_length로 나눠 평균 사용된 page frame을 계산한다.
- 4. 프로그램에서 사용된 array는 모두 동적 메모리 할당을 받았다.
  - 5. array는 pointer의 형태로 argument로 전달된다. 메모리 측면에서 array 전체를 복사하여 전달하는 것보다 효율적일 뿐 아니라 함수 내에서 값을 변경할 때도 용이하기 때문이다.
  - 6. LFU의 경우 같은 값을 갖는 page에 한해 LRU기법을 적용해야 한다. 따라서 LRU 기법을 적용할 때 사용할 used time을 저장한 array 'tie\_break'를 LFU\_test 함수 내에서 정의하고 사용한다. Figure 5에서 보이는 것처럼 mark 내에서 최솟값(가장 작은 used count)을 찾은 후 이와 같은 값을 갖는 page에 한해 tie\_break에서 가장 작은 used time을 갖는 page를 찾는다.

```

int LFU_replace(int *mem, int *mark, int *tie_break, int page_frame){
    int min = mark[0];
    int min_no = 0;
    for(int i = 0; i < page_frame; i++){
        if(min > mark[i]){
            min = mark[i];
        }
    }
    int tie_min = tie_break[0];
    // if the page has same used count with minimum value, tie_break by LRU
    for(int i = 0; i < page_frame; i++){
        if(mark[i] == min){
            if(tie_min > tie_break[i]){
                tie_min = tie_break[i];
                min_no = i;
            }
        }
    }
    return min_no;
}

```

Figure 5 LFU tie break

- replace할 page를 고를 때 rule을 만족하는 page가 1개 이상 존재하면 page frame의 index가 작은 page를 선택한다. (LFU의 경우 LRU를 이용하여 page를 선정할 때 같은 used time을 가지는 page에 대해 적용된다.)

## Result

### Error test 및 예외 처리

- 파일이 존재하지 않을 때

```

dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ls
OS2020-2_2018312292_김동원_P3.cbp  OS2020-2_2018312292_김동원_P3.layout  error1.txt  error3.txt  error5.txt  main  obj  rand_input.txt
OS2020-2_2018312292_김동원_P3.depend  bin  error2.txt  error4.txt  input1.txt  main.c  output.txt
dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: error
[Error] File opening failed
dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$

```

Figure 6 파일이 존재하지 않음

- reference string에서 주어진 page 번호가 범위를 벗어날 때

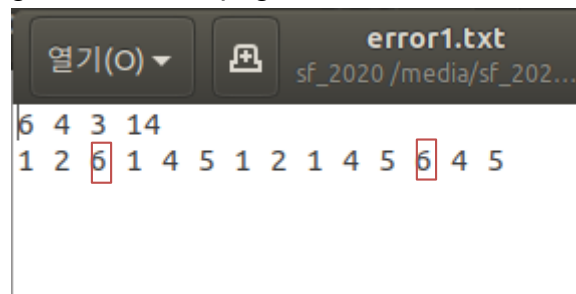



Figure 7 page 범위 벗어남

```

dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: error1.txt
6 4 3 14
1 2 6 1 4 5 1 2 1 4 5 6 4 5
[Error] maximum page number can't be bigger than total page no.

```

Figure 8 page 범위 벗어남 에러 문구

열기(O) ▼  error2.txt  
sf\_2020 /media/sf\_202...


6 4 3 14  
1 -1 5 1 4 5 1 2 1 4 5 2 4 5

Figure 9 page 범위 벗어남(마이너스 범위)

```
dw@dw-VirtualBox:/media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: error2.txt
6 4 3 14
1 -1 5 1 4 5 1 2 1 4 5 2 4 5
[Error] page number can't be smaller than 0
```

Figure 10 page 범위 벗어남(마이너스 범위) 에러 문구

- reference string의 길이가 명시된 것보다 작음

열기(O) ▼  error4.txt  
sf\_2020 /media/sf\_202...


6 4 3 14  
1 2 5 1 4 5 1 2 1 4 5 2

Figure 11 reference string 짧음

```
dw@dw-VirtualBox:/media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: error4.txt
6 4 3 14
1 2 5 1 4 5 1 2 1 4 5 2 0 0
[Error] no. of page reference and string's length not matched
```

Figure 12 reference string 짧음 에러 문구

- reference string이 명시된 것보다 김

열기(O) ▼  error3.txt  
sf\_2020 /media/sf\_202..

6 4 3 14  
1 2 5 1 4 5 1 2 1 4 5 2 4 5 4

Figure 13 reference string 김(len: 15)

```

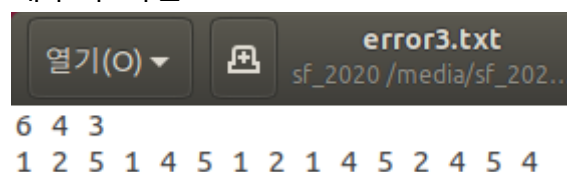
dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_p3$ ./main
Enter 'random'(random input) or file name to open: error3.txt
6 4 3 14
1 2 5 1 4 5 1 2 1 4 5 2 4 5
--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--LRU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--LFU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--Clock--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--WS--
<page fault> Time 1 Insert 1 no. of fault 1
<page fault> Time 2 Insert 2 no. of fault 2
<page fault> Time 3 Insert 5 no. of fault 3
<page fault> Time 5 Insert 4 no. of fault 4
<page fault> Time 8 Insert 2 no. of fault 5
<page fault> Time 10 Insert 4 no. of fault 6
<page fault> Time 11 Insert 5 no. of fault 7
average page frame no.: 2.93
Total no. of fault: 7/14

```

Figure 14 명시된 것보다 긴 부분은 무시하고 진행

명시된 길이는 14인 것에 비해 실제로 input file에는 reference string가 15이다. 이런 경우 명시된 길이보다 긴 부분은 무시하고 시뮬레이션을 진행한다.

##### 5. 첫 줄의 정수 개수가 작음



```

6 4 3
1 2 5 1 4 5 1 2 1 4 5 2 4 5 4

```

Figure 15 첫 줄 정수 개수 작음(3개)

```

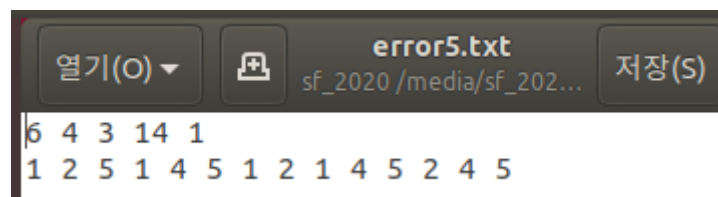
dw@dw-VirtualBox:/media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: error3.txt
6 4 3 1
2
--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
Total no. of fault: 1/1
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
Total no. of fault: 1/1
--LRU--
<page fault> Time 1 replaced index 0 no. of fault 1
Total no. of fault: 1/1
--LFU--
<page fault> Time 1 replaced index 0 no. of fault 1
Total no. of fault: 1/1
--Clock--
<page fault> Time 1 replaced index 0 no. of fault 1
Total no. of fault: 1/1
--WS--
<page fault> Time 1 Insert 2 no. of fault 1
average page frame no.: 5.00
Total no. of fault: 1/1

```

Figure 16 줄에 상관없이 처음 등장하느 정수 4개 사용

첫 줄 정수 개수가 4보다 작으면 다음 줄까지 합쳐 4개를 채운 후 이 정보를 이용하여 시뮬레이션을 진행한다.

#### 6. 첫 줄 정수 개수가 많음



```

6 4 3 14 1
1 2 5 1 4 5 1 2 1 4 5 2 4 5

```

Figure 17 첫 줄 정수 개수 많음 (5개)

```

dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: error5.txt
6 4 3 14
1 1 2 5 1 4 5 1 2 1 4 5 2 4
--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 3 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
<page fault> Time 6 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 3 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
<page fault> Time 6 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--LRU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 3 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
<page fault> Time 6 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--LFU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 3 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
<page fault> Time 6 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--Clock--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 3 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
<page fault> Time 6 replaced index 3 no. of fault 4
Total no. of fault: 4/14
--WS--
<page fault> Time 1 Insert 1 no. of fault 1
<page fault> Time 3 Insert 2 no. of fault 2
<page fault> Time 4 Insert 5 no. of fault 3
<page fault> Time 6 Insert 4 no. of fault 4
<page fault> Time 9 Insert 2 no. of fault 5
<page fault> Time 11 Insert 4 no. of fault 6
<page fault> Time 12 Insert 5 no. of fault 7
average page frame no.: 3.07
Total no. of fault: 7/14

```

Figure 18 첫 줄 정수 개수 5개 -4개를 제외한 값은 reference string의 일부로 취급

## input file을 이용한 test

```
dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: input1.txt
6 4 3 14
0 1 5 0 3 4 0 1 0 3 4 5 3 4
--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
<page fault> Time 6 replaced index 2 no. of fault 5
<page fault> Time 12 replaced index 0 no. of fault 6
Total no. of fault: 6/14
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
<page fault> Time 6 replaced index 0 no. of fault 5
<page fault> Time 7 replaced index 1 no. of fault 6
<page fault> Time 8 replaced index 2 no. of fault 7
<page fault> Time 12 replaced index 3 no. of fault 8
<page fault> Time 13 replaced index 0 no. of fault 9
<page fault> Time 14 replaced index 1 no. of fault 10
Total no. of fault: 10/14
--LRU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
<page fault> Time 6 replaced index 1 no. of fault 5
<page fault> Time 8 replaced index 2 no. of fault 6
<page fault> Time 12 replaced index 2 no. of fault 7
Total no. of fault: 7/14
--LFU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
<page fault> Time 6 replaced index 1 no. of fault 5
<page fault> Time 8 replaced index 2 no. of fault 6
<page fault> Time 12 replaced index 2 no. of fault 7
Total no. of fault: 7/14
```



```

--Clock--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 5 replaced index 3 no. of fault 4
<page fault> Time 6 replaced index 0 no. of fault 5
<page fault> Time 7 replaced index 1 no. of fault 6
<page fault> Time 8 replaced index 2 no. of fault 7
<page fault> Time 12 replaced index 3 no. of fault 8
<page fault> Time 13 replaced index 0 no. of fault 9
<page fault> Time 14 replaced index 1 no. of fault 10
Total no. of fault: 10/14
--WS--
<page fault> Time 1 Insert 0 no. of fault 1
<page fault> Time 2 Insert 1 no. of fault 2
<page fault> Time 3 Insert 5 no. of fault 3
<page fault> Time 5 Insert 3 no. of fault 4
<page fault> Time 6 Insert 4 no. of fault 5
<page fault> Time 8 Insert 1 no. of fault 6
<page fault> Time 10 Insert 3 no. of fault 7
<page fault> Time 11 Insert 4 no. of fault 8
<page fault> Time 12 Insert 5 no. of fault 9
average page frame no.: 3.14
Total no. of fault: 9/14

```

Figure 19 input1.txt console 결과

test의 결과는 console 출력과 output 파일, 두 가지이다. 콘솔의 첫 줄에는 page개수, page frame 개수, window 크기, reference string의 길이가 표현된다. 두 번째 줄에는 reference string으로 사용된 정수들이 나타난다. page fault가 발생했을 때 그 시간, page frame에서 victim이 된 page의 page frame index(FA) 혹은 memory에 load 된 page의 번호(VA), 현재까지 page fault의 횟수가 표시된다. 모든 reference string을 다 검사한 후 총 reference 중 page fault의 비율을 출력한다. WS 경우 평균 page frame의 개수도 표현한다.

```
output.txt
sf_2020 /media/sf_2020/운...
열기(O)  저장(S)  ≡  -  +  x

--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
<page fault> Time 2 replaced index 1 no. of fault 2
[Time 2] 0 1
<page fault> Time 3 replaced index 2 no. of fault 3
[Time 3] 0 1 5
[Time 4] 0 1 5
<page fault> Time 5 replaced index 3 no. of fault 4
[Time 5] 0 1 5 3
<page fault> Time 6 replaced index 2 no. of fault 5
[Time 6] 0 1 4 3
[Time 7] 0 1 4 3
[Time 8] 0 1 4 3
[Time 9] 0 1 4 3
[Time 10] 0 1 4 3
[Time 11] 0 1 4 3
<page fault> Time 12 replaced index 0 no. of fault 6
[Time 12] 5 1 4 3
[Time 13] 5 1 4 3
[Time 14] 5 1 4 3
Total no. of fault: 6/14
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
<page fault> Time 2 replaced index 1 no. of fault 2
[Time 2] 0 1
<page fault> Time 3 replaced index 2 no. of fault 3
[Time 3] 0 1 5
[Time 4] 0 1 5
<page fault> Time 5 replaced index 3 no. of fault 4
[Time 5] 0 1 5 3
<page fault> Time 6 replaced index 0 no. of fault 5
[Time 6] 4 1 5 3
<page fault> Time 7 replaced index 1 no. of fault 6
[Time 7] 4 0 5 3
<page fault> Time 8 replaced index 2 no. of fault 7
[Time 8] 4 0 1 3
[Time 9] 4 0 1 3
[Time 10] 4 0 1 3
[Time 11] 4 0 1 3 |
<page fault> Time 12 replaced index 3 no. of fault 8
[Time 12] 4 0 1 5
<page fault> Time 13 replaced index 0 no. of fault 9
[Time 13] 3 0 1 5
<page fault> Time 14 replaced index 1 no. of fault 10
[Time 14] 3 4 1 5
```

---

Total no. of fault: 10/14

--LRU--

<page fault> Time 1 replaced index 0 no. of fault 1

[Time 1] 0

<page fault> Time 2 replaced index 1 no. of fault 2

[Time 2] 0 1

<page fault> Time 3 replaced index 2 no. of fault 3

[Time 3] 0 1 5

[Time 4] 0 1 5

<page fault> Time 5 replaced index 3 no. of fault 4

[Time 5] 0 1 5 3

<page fault> Time 6 replaced index 1 no. of fault 5

[Time 6] 0 4 5 3

[Time 7] 0 4 5 3

<page fault> Time 8 replaced index 2 no. of fault 6

[Time 8] 0 4 1 3

[Time 9] 0 4 1 3

[Time 10] 0 4 1 3

[Time 11] 0 4 1 3

<page fault> Time 12 replaced index 2 no. of fault 7

[Time 12] 0 4 5 3

[Time 13] 0 4 5 3

[Time 14] 0 4 5 3

Total no. of fault: 7/14

--LFU--

<page fault> Time 1 replaced index 0 no. of fault 1

[Time 1] 0

<page fault> Time 2 replaced index 1 no. of fault 2

[Time 2] 0 1

<page fault> Time 3 replaced index 2 no. of fault 3

[Time 3] 0 1 5

[Time 4] 0 1 5

<page fault> Time 5 replaced index 3 no. of fault 4

[Time 5] 0 1 5 3

<page fault> Time 6 replaced index 1 no. of fault 5

[Time 6] 0 4 5 3

[Time 7] 0 4 5 3

<page fault> Time 8 replaced index 2 no. of fault 6

[Time 8] 0 4 1 3

[Time 9] 0 4 1 3

[Time 10] 0 4 1 3

[Time 11] 0 4 1 3

<page fault> Time 12 replaced index 2 no. of fault 7

[Time 12] 0 4 5 3

[Time 13] 0 4 5 3

[Time 14] 0 4 5 3

Total no. of fault: 7/14

--Clock--

```
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
<page fault> Time 2 replaced index 1 no. of fault 2
[Time 2] 0 1
<page fault> Time 3 replaced index 2 no. of fault 3
[Time 3] 0 1 5
[Time 4] 0 1 5
<page fault> Time 5 replaced index 3 no. of fault 4
[Time 5] 0 1 5 3
<page fault> Time 6 replaced index 0 no. of fault 5
[Time 6] 4 1 5 3
<page fault> Time 7 replaced index 1 no. of fault 6
[Time 7] 4 0 5 3
<page fault> Time 8 replaced index 2 no. of fault 7
[Time 8] 4 0 1 3
[Time 9] 4 0 1 3
[Time 10] 4 0 1 3
[Time 11] 4 0 1 3
<page fault> Time 12 replaced index 3 no. of fault 8
[Time 12] 4 0 1 5
<page fault> Time 13 replaced index 0 no. of fault 9
[Time 13] 3 0 1 5
<page fault> Time 14 replaced index 1 no. of fault 10
[Time 14] 3 4 1 5
Total no. of fault: 10/14
```

--WS--

```
<page fault> Time 1 Insert 0 no. of fault 1
[Time 1] 0
<page fault> Time 2 Insert 1 no. of fault 2
[Time 2] 0 1
<page fault> Time 3 Insert 5 no. of fault 3
[Time 3] 0 1 5
[Time 4] 0 1 5
<page fault> Time 5 Insert 3 no. of fault 4
[Time 5] 0 1 3 5
<page fault> Time 6 Insert 4 no. of fault 5
[Time 6] 0 3 4 5
[Time 7] 0 3 4
<page fault> Time 8 Insert 1 no. of fault 6
[Time 8] 0 1 3 4
[Time 9] 0 1 4
<page fault> Time 10 Insert 3 no. of fault 7
[Time 10] 0 1 3
<page fault> Time 11 Insert 4 no. of fault 8
[Time 11] 0 1 3 4
<page fault> Time 12 Insert 5 no. of fault 9
```

```
[Time 12] 0 3 4 5
[Time 13] 3 4 5
[Time 14] 3 4 5
average page frame no.: 3.14
Total no. of fault: 9/14
```

Figure 20 input1.txt output file

output file은 시간 별로 residence set을 추가로 표시한다.

```
dw@dw-VirtualBox:/media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: input10.txt
6 3 3 12
0 1 2 3 0 1 4 0 1 2 3 4
--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 4 replaced index 2 no. of fault 4
<page fault> Time 7 replaced index 2 no. of fault 5
<page fault> Time 10 replaced index 0 no. of fault 6
<page fault> Time 11 replaced index 0 no. of fault 7
Total no. of fault: 7/12
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 4 replaced index 0 no. of fault 4
<page fault> Time 5 replaced index 1 no. of fault 5
<page fault> Time 6 replaced index 2 no. of fault 6
<page fault> Time 7 replaced index 0 no. of fault 7
<page fault> Time 10 replaced index 1 no. of fault 8
<page fault> Time 11 replaced index 2 no. of fault 9
Total no. of fault: 9/12
```

Figure 21 frame 3일 때

```

dw@dw-VirtualBox:/media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: input11.txt
6 4 3 12
0 1 2 3 0 1 4 0 1 2 3 4
--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 4 replaced index 3 no. of fault 4
<page fault> Time 7 replaced index 3 no. of fault 5
<page fault> Time 11 replaced index 0 no. of fault 6
Total no. of fault: 6/12
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 3 replaced index 2 no. of fault 3
<page fault> Time 4 replaced index 3 no. of fault 4
<page fault> Time 7 replaced index 0 no. of fault 5
<page fault> Time 8 replaced index 1 no. of fault 6
<page fault> Time 9 replaced index 2 no. of fault 7
<page fault> Time 10 replaced index 3 no. of fault 8
<page fault> Time 11 replaced index 0 no. of fault 9
<page fault> Time 12 replaced index 1 no. of fault 10
Total no. of fault: 10/12

```

Figure 22 frame 4일 때

FIFO의 경우 수업에서 배운 대로 frame의 개수(3 -> 4)가 증가했음에도 fault 횟수가 오히려 증가(9 -> 10)하는 Belady's anomaly가 발생하는 것을 확인할 수 있다.

#### random input

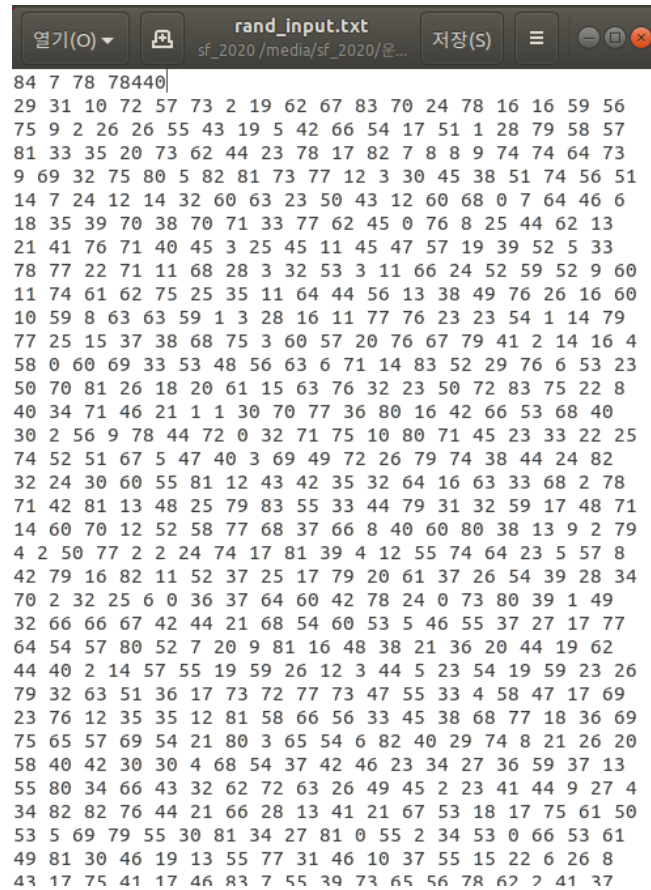
1. page의 범위: 1 ~ PAGE
2. page\_frame의 범위: 1 ~ FRAME
3. window size 크기의 범위: 1 ~ WINDOW

window size를 page개수보다 작게 설정해야 하는지 고민하기도 하였으나 page 개수보다 큰 window size를 가지는 것 또한 의미 있는 시뮬레이션 결과가 될 것 같아 이와 같이 설정하였다.

4. reference string length의 범위: page + 1 ~ S\_LEN

다양한 page가 reference될 수 있도록 모든 page를 사용하지는 않더라도 적어도 page 수보다는 많은 reference를 테스트할 수 있도록 이와 같이 설정하였다.





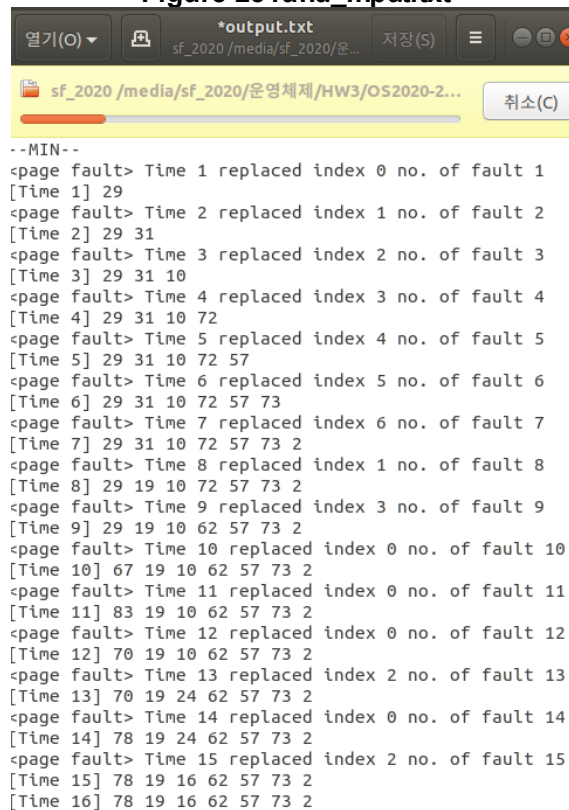
```

rand_input.txt
sf_2020 /media/sf_2020/문...
저장(S)

84 7 78 78440
29 31 10 72 57 73 2 19 62 67 83 70 24 78 16 16 59 56
75 9 2 26 26 55 43 19 5 42 66 54 17 51 1 28 79 58 57
81 33 35 20 73 62 44 23 78 17 82 7 8 8 9 74 74 64 73
9 69 32 75 80 5 82 81 73 77 12 3 30 45 38 51 74 56 51
14 7 24 12 14 32 60 63 23 50 43 12 60 68 0 7 64 46 6
18 35 39 70 38 70 71 33 77 62 45 0 76 8 25 44 62 13
21 41 76 71 40 45 3 25 45 11 45 47 57 19 39 52 5 33
78 77 22 71 11 68 28 3 32 53 3 11 66 24 52 59 52 9 60
11 74 61 62 75 25 35 11 64 44 56 13 38 49 76 26 16 60
10 59 8 63 63 59 1 3 28 16 11 77 76 23 23 54 1 14 79
77 25 15 37 38 68 75 3 60 57 20 76 67 79 41 2 14 16 4
58 0 60 69 33 53 48 56 63 6 71 14 83 52 29 76 6 53 23
50 70 81 26 18 20 61 15 63 76 32 23 50 72 83 75 22 8
40 34 71 46 21 1 1 30 70 77 36 80 16 42 66 53 68 40
30 2 56 9 78 44 72 0 32 71 75 10 80 71 45 23 33 22 25
74 52 51 67 5 47 40 3 69 49 72 26 79 74 38 44 24 82
32 24 30 60 55 81 12 43 42 35 32 64 16 63 33 68 2 78
71 42 81 13 48 25 79 83 55 33 44 79 31 32 59 17 48 71
14 60 70 12 52 58 77 68 37 66 8 40 60 80 38 13 9 2 79
4 2 50 77 2 2 24 74 17 81 39 4 12 55 74 64 23 5 57 8
42 79 16 82 11 52 37 25 17 79 20 61 37 26 54 39 28 34
70 2 32 25 6 0 36 37 64 60 42 78 24 0 73 80 39 1 49
32 66 66 67 42 44 21 68 54 60 53 5 46 55 37 27 17 77
64 54 57 80 52 7 20 9 81 16 48 38 21 36 20 44 19 62
44 40 2 14 57 55 19 59 26 12 3 44 5 23 54 19 59 23 26
79 32 63 51 36 17 73 72 77 73 47 55 33 4 58 47 17 69
23 76 12 35 35 12 81 58 66 56 33 45 38 68 77 18 36 69
75 65 57 69 54 21 80 3 65 54 6 82 40 29 74 8 21 26 20
58 40 42 30 30 4 68 54 37 42 46 23 34 27 36 59 37 13
55 80 34 66 43 32 62 72 63 26 49 45 2 23 41 44 9 27 4
34 82 82 76 44 21 66 28 13 41 21 67 53 18 17 75 61 50
53 5 69 79 55 30 81 34 27 81 0 55 2 34 53 0 66 53 61
49 81 30 46 19 13 55 77 31 46 10 37 55 15 22 6 26 8
43 17 75 41 17 46 83 7 55 39 73 65 56 78 62 2 41 37

```

Figure 23 rand\_input.txt



```

*output.txt
sf_2020 /media/sf_2020/문...
저장(S)

sf_2020 /media/sf_2020/운영체제/HW3/OS2020-2... 취소(C)

--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 29
<page fault> Time 2 replaced index 1 no. of fault 2
[Time 2] 29 31
<page fault> Time 3 replaced index 2 no. of fault 3
[Time 3] 29 31 10
<page fault> Time 4 replaced index 3 no. of fault 4
[Time 4] 29 31 10 72
<page fault> Time 5 replaced index 4 no. of fault 5
[Time 5] 29 31 10 72 57
<page fault> Time 6 replaced index 5 no. of fault 6
[Time 6] 29 31 10 72 57 73
<page fault> Time 7 replaced index 6 no. of fault 7
[Time 7] 29 31 10 72 57 73 2
<page fault> Time 8 replaced index 1 no. of fault 8
[Time 8] 29 19 10 72 57 73 2
<page fault> Time 9 replaced index 3 no. of fault 9
[Time 9] 29 19 10 62 57 73 2
<page fault> Time 10 replaced index 0 no. of fault 10
[Time 10] 67 19 10 62 57 73 2
<page fault> Time 11 replaced index 0 no. of fault 11
[Time 11] 83 19 10 62 57 73 2
<page fault> Time 12 replaced index 0 no. of fault 12
[Time 12] 70 19 10 62 57 73 2
<page fault> Time 13 replaced index 2 no. of fault 13
[Time 13] 70 19 24 62 57 73 2
<page fault> Time 14 replaced index 0 no. of fault 14
[Time 14] 78 19 24 62 57 73 2
<page fault> Time 15 replaced index 2 no. of fault 15
[Time 15] 78 19 16 62 57 73 2
[Time 16] 78 19 16 62 57 73 2

```

Figure 24 rand\_input.txt의 결과

Figure 21, 22에서 볼 수 있듯이 test 내용이 너무 많아 적절한 test가 이뤄졌는지

확인하기 힘들기 때문에 보고서 작성을 위해 range를 잠시 조정하도록 한다.

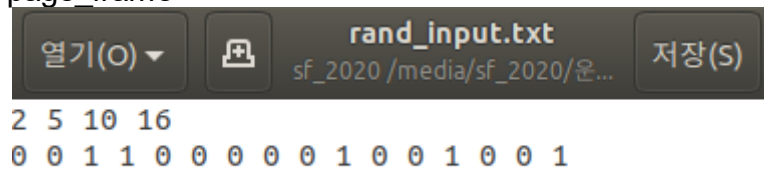
이렇게 긴 reference string을 test할 때도 dynamic memory allocation 및 array pointer를 argument로 이용하기 때문에 불필요한 메모리 낭비를 줄일 수 있다.

```
#define PAGE 10           // limit of no. of page
#define FRAME 5           // limit of no. of page frame allocated
#define WINDOW 10        // limit of window size
#define S_LEN 50         // limit of length of reference string
```

Figure 25 조정된 range

다음은 random으로 test한 case 중 일부이다.

1.  $\text{page} < \text{page\_frame}$



rand\_input.txt  
sf\_2020 /media/sf\_2020/운... 저장(S)

2 5 10 16  
0 0 1 1 0 0 0 0 0 1 0 0 1 0 0 1

Figure 26 page 개수 < page frame 개수 input(input6.txt에 저장)



```
output.txt
sf_2020 /media/sf_2020/운...
열기(O)  저장(S)  ≡

--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
[Time 2] 0
<page fault> Time 3 replaced index 1 no. of fault 2
[Time 3] 0 1
[Time 4] 0 1
[Time 5] 0 1
[Time 6] 0 1
[Time 7] 0 1
[Time 8] 0 1
[Time 9] 0 1
[Time 10] 0 1
[Time 11] 0 1
[Time 12] 0 1
[Time 13] 0 1
[Time 14] 0 1
[Time 15] 0 1
[Time 16] 0 1
Total no. of fault: 2/16
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
[Time 2] 0
<page fault> Time 3 replaced index 1 no. of fault 2
[Time 3] 0 1
[Time 4] 0 1
[Time 5] 0 1
[Time 6] 0 1
[Time 7] 0 1
[Time 8] 0 1
[Time 9] 0 1
[Time 10] 0 1
[Time 11] 0 1
[Time 12] 0 1
[Time 13] 0 1
[Time 14] 0 1
[Time 15] 0 1
[Time 16] 0 1
Total no. of fault: 2/16
--LRU--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
[Time 2] 0
<page fault> Time 3 replaced index 1 no. of fault 2
[Time 3] 0 1
[Time 4] 0 1
[Time 5] 0 1
[Time 6] 0 1
[Time 7] 0 1
```

열기(O) ▾



output.txt

sf\_2020 /media/sf\_2020/운...

저장(S)



```
[Time 8] 0 1
[Time 9] 0 1
[Time 10] 0 1
[Time 11] 0 1
[Time 12] 0 1
[Time 13] 0 1
[Time 14] 0 1
[Time 15] 0 1
[Time 16] 0 1
Total no. of fault: 2/16
--LFU--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
[Time 2] 0
<page fault> Time 3 replaced index 1 no. of fault 2
[Time 3] 0 1
[Time 4] 0 1
[Time 5] 0 1
[Time 6] 0 1
[Time 7] 0 1
[Time 8] 0 1
[Time 9] 0 1
[Time 10] 0 1
[Time 11] 0 1
[Time 12] 0 1
[Time 13] 0 1
[Time 14] 0 1
[Time 15] 0 1
[Time 16] 0 1
Total no. of fault: 2/16
--Clock--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 0
[Time 2] 0
<page fault> Time 3 replaced index 1 no. of fault 2
[Time 3] 0 1
[Time 4] 0 1
[Time 5] 0 1
[Time 6] 0 1
[Time 7] 0 1
[Time 8] 0 1
[Time 9] 0 1
[Time 10] 0 1
[Time 11] 0 1
[Time 12] 0 1
[Time 13] 0 1
[Time 14] 0 1
[Time 15] 0 1
[Time 16] 0 1
Total no. of fault: 2/16
```

```

--WS--
<page fault> Time 1 Insert 0 no. of fault 1
[Time 1] 0
[Time 2] 0
<page fault> Time 3 Insert 1 no. of fault 2
[Time 3] 0 1
[Time 4] 0 1
[Time 5] 0 1
[Time 6] 0 1
[Time 7] 0 1
[Time 8] 0 1
[Time 9] 0 1
[Time 10] 0 1
[Time 11] 0 1
[Time 12] 0 1
[Time 13] 0 1
[Time 14] 0 1
[Time 15] 0 1
[Time 16] 0 1
average page frame no.: 1.88
Total no. of fault: 2/16

```

Figure 27 page 개수 < page frame 개수 output

```

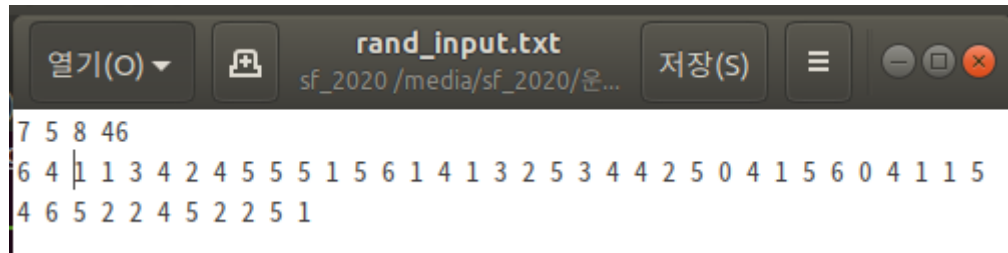
dw@dw-VirtualBox: /media/sf_2020/운영체제/HW3/OS2020-2_2018312292_김동원_P3$ ./main
Enter 'random'(random input) or file name to open: input7.txt
3 4 4 11
0 1 1 2 1 2 1 1 2 1 0
--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
Total no. of fault: 3/11
--FIFO--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
Total no. of fault: 3/11
--LRU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
Total no. of fault: 3/11
--LFU--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
Total no. of fault: 3/11
--Clock--
<page fault> Time 1 replaced index 0 no. of fault 1
<page fault> Time 2 replaced index 1 no. of fault 2
<page fault> Time 4 replaced index 2 no. of fault 3
Total no. of fault: 3/11
--WS--
<page fault> Time 1 Insert 0 no. of fault 1
<page fault> Time 2 Insert 1 no. of fault 2
<page fault> Time 4 Insert 2 no. of fault 3
<page fault> Time 11 Insert 0 no. of fault 4
average page frame no.: 2.18
Total no. of fault: 4/11

```

Figure 28 WS 취약점(input7.txt에 저장)

Figure 26에서도 볼 수 있듯이 page의 개수가 할당 받은 frame의 개수보다 작아서 FA의 경우 첫 load에만 fault가 발생하는 반면 VA인 WS는 working set만 frame내에 존재하기 때문에 FA보다 fault가 많이 발생하기도 한다.

## 2. 일반적인 상황

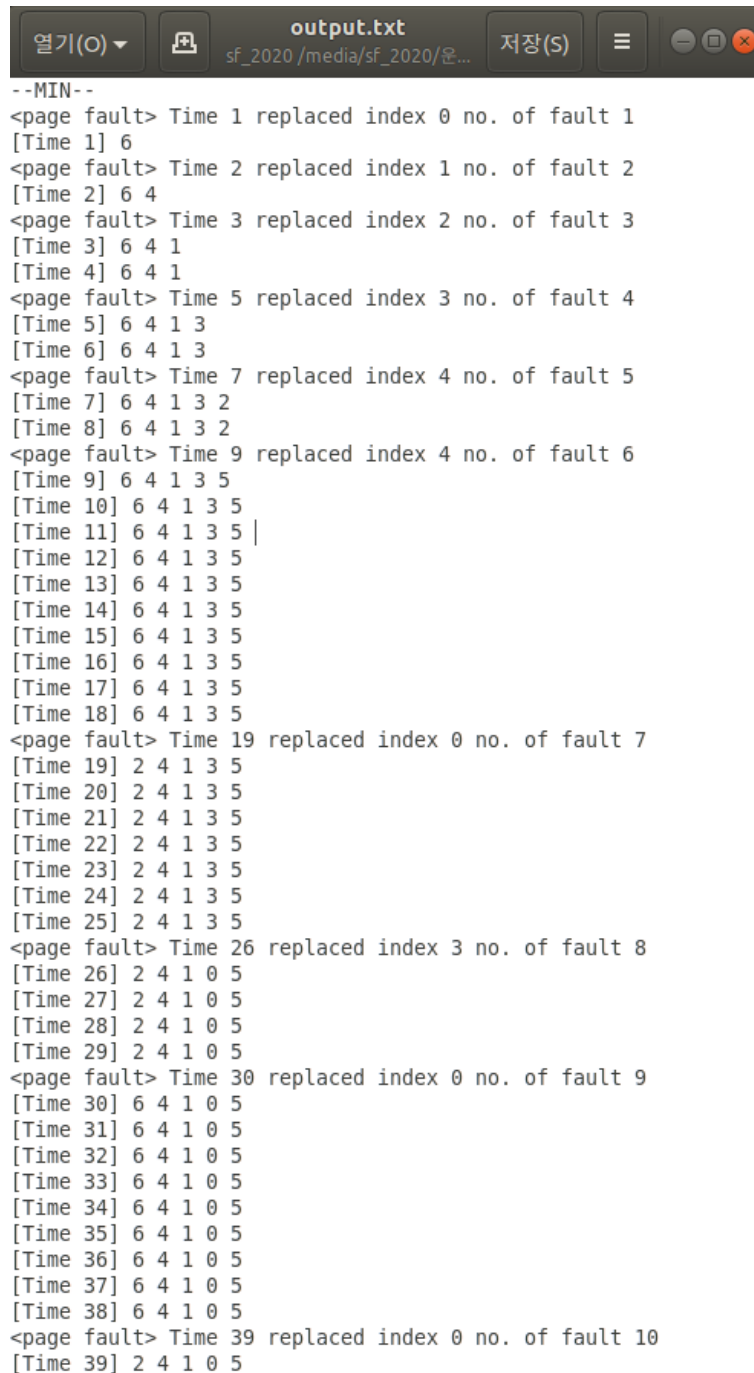


```

7 5 8 46
6 4 1 1 3 4 2 4 5 5 5 1 5 6 1 4 1 3 2 5 3 4 4 2 5 0 4 1 5 6 0 4 1 1 5
4 6 5 2 2 4 5 2 2 5 1

```

Figure 29 일반적인 random input(input8.txt에 저장)



```

--MIN--
<page fault> Time 1 replaced index 0 no. of fault 1
[Time 1] 6
<page fault> Time 2 replaced index 1 no. of fault 2
[Time 2] 6 4
<page fault> Time 3 replaced index 2 no. of fault 3
[Time 3] 6 4 1
[Time 4] 6 4 1
<page fault> Time 5 replaced index 3 no. of fault 4
[Time 5] 6 4 1 3
[Time 6] 6 4 1 3
<page fault> Time 7 replaced index 4 no. of fault 5
[Time 7] 6 4 1 3 2
[Time 8] 6 4 1 3 2
<page fault> Time 9 replaced index 4 no. of fault 6
[Time 9] 6 4 1 3 5
[Time 10] 6 4 1 3 5
[Time 11] 6 4 1 3 5
[Time 12] 6 4 1 3 5
[Time 13] 6 4 1 3 5
[Time 14] 6 4 1 3 5
[Time 15] 6 4 1 3 5
[Time 16] 6 4 1 3 5
[Time 17] 6 4 1 3 5
[Time 18] 6 4 1 3 5
<page fault> Time 19 replaced index 0 no. of fault 7
[Time 19] 2 4 1 3 5
[Time 20] 2 4 1 3 5
[Time 21] 2 4 1 3 5
[Time 22] 2 4 1 3 5
[Time 23] 2 4 1 3 5
[Time 24] 2 4 1 3 5
[Time 25] 2 4 1 3 5
<page fault> Time 26 replaced index 3 no. of fault 8
[Time 26] 2 4 1 0 5
[Time 27] 2 4 1 0 5
[Time 28] 2 4 1 0 5
[Time 29] 2 4 1 0 5
<page fault> Time 30 replaced index 0 no. of fault 9
[Time 30] 6 4 1 0 5
[Time 31] 6 4 1 0 5
[Time 32] 6 4 1 0 5
[Time 33] 6 4 1 0 5
[Time 34] 6 4 1 0 5
[Time 35] 6 4 1 0 5
[Time 36] 6 4 1 0 5
[Time 37] 6 4 1 0 5
[Time 38] 6 4 1 0 5
<page fault> Time 39 replaced index 0 no. of fault 10
[Time 39] 2 4 1 0 5

```

Figure 29 Figure27 결과 중 일부

결과가 많아서 최종적인 page fault 개수만 나타낸다.

MIN	10/46
FIFO	18/46
LRU	14/46
LFU	13/46
Clock	17/46
WS	14/46 (avg frame: 4.83)

Table 1 Figure27 결과

3. page 개수가 page frame보다 과하게 많을 때

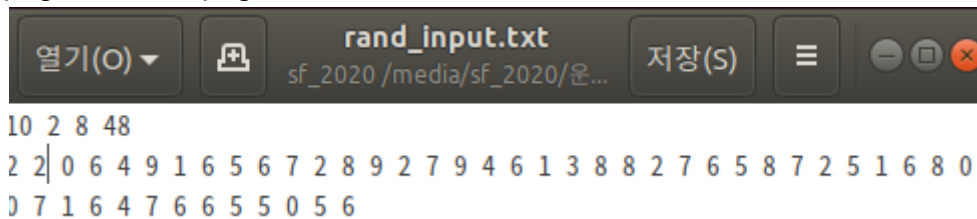


Figure 30 page >> frame input(input9.txt에 저장)

MIN	33/48
FIFO	41/48
LRU	41/48
LFU	41/48
Clock	41/48
WS	19/48 (avg. frame: 6.00)

Table 2 page >> frame output

보다시피 page가 frame보다 과도하게 많은 경우 FA는 대부분 page fault가 발생한다. 그에 반해 WS는 window 사이즈에 의해 영향을 받기 때문에 다른 기법에 비해 page fault가 매우 작은 것을 확인할 수 있다.

4. window 사이즈에 따른 WS 결과 분석

Table 2에서 볼 수 있듯이 WS는 window 사이즈에 의해 영향을 받으므로 임의로 만든 input에서 window 사이즈만 변경시켜 결과를 분석해본다. window 사이즈가 클수록 많은 수의 page가 frame 내에 존재할 가능성이 높아지므로 fault수는 감소하는 것에 반해 평균 page frame의 수는 증가할 것으로 예측된다.



Figure 31 비교할 random input(input5.txt에 저장)

MIN	16/22
FIFO	17/22
LRU	19/22
LFU	21/22
Clock	17/22
WS	12/22 (avg. frame: 4.64)

Table 3 비교할 random output

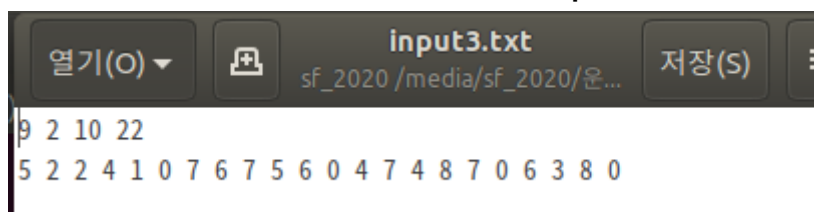


Figure 32 window size를 10으로 증가(input3.txt)

MIN	16/22
FIFO	17/22
LRU	19/22
LFU	21/22
Clock	17/22
WS	9/22 (avg. frame: 5.45)

Table 4 window size 10일 때

window의 크기가 크게 설정되니 (page 수에 가깝게 설정될수록) fault가 줄어든다. 이 경우 page 수보다 window의 크기가 더 크도록 설정했으므로 더더욱 그럴 것이다.



Figure 33 window size 2로 감소(input4.txt)

MIN	16/22
FIFO	17/22
LRU	19/22
LFU	21/22
Clock	17/22
WS	17/22 (avg. frame: 2.68)

Table 5 window size 2일 때

예측한대로 window size가 커지면 fault가 감소하는 대신 평균 frame의 수가 많아지고 window size가 작아지면 fault가 증가하는 대신 평균 frame의 수가 작아진다. Table5의 결과에서 FA는 page frame 수 2, VA는 2.68로 비슷하다고 볼 수 있다.

그러나 WS는 다른 기법에 비해 MIN에 가까운 결과를 내놓으므로 비교적 우수한 기법이라고 할 수 있다. (물론 MIN 기법 자체가 FA 중 optimal한 기법이기에 때문에 이를 VA 기법 중 하나인 WS의 결과와 비교하는 것은 부적절하다고 볼 수 있으나 WS의 평균 page frame 수가 FA와 비슷하기 때문에 비교를 진행해봤다.)

이 외에도 다양한 random input을 직접 프로그램을 이용하여 테스트할 수 있다. (범위는 과제에서 주어진 대로 설정되어 있다.)

### **C. Conclusion**

page와 page frame간의 관계로 인해 FA의 page fault 횟수가 결정되는 것을 확인할 수 있었다. page가 frame보다 지나치게 많으면 page fault가 과도하게 발생하고 page보다 frame이 많으면 load할 때만 fault가 발생한다. WS의 경우 window의 크기가 크면 fault는 줄어드는 추세이지만 평균 frame의 개수는 많아지고 window의 크기가 작아지면 반대가 된다. 따라서 성능 향상을 위해서는 적절한 page frame과 window 크기를 구하는 것이 중요해 보인다.

### **D. Reference**

- [1] 엄영익, "운영체제 Project-3 [Virtual Memory Management 기법 구현]," 수원, 2020.