개요

본 프로그램은 표준 입력함수를 통해 console에서 process에 대한 정보와 몇 가지 제약 조건을 받아, Demand paging system에서 replacement 기법에 따른 시뮬레이션 결과를 다음과 같은 순서로 출력한다.

* MIN, FIFO, LRU, LFU, WS 기법을 적용했을때,
  + 메모리 상태 변화 과정
  + page fault 발생 횟수 및 발생 위치

입력

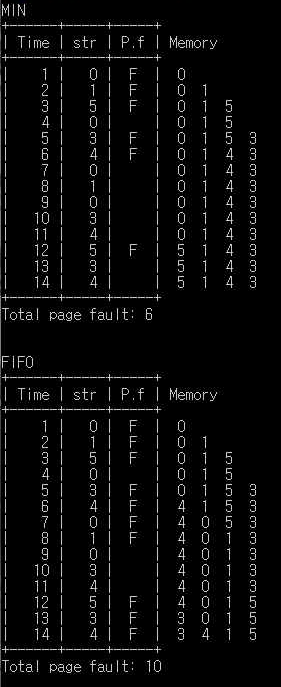
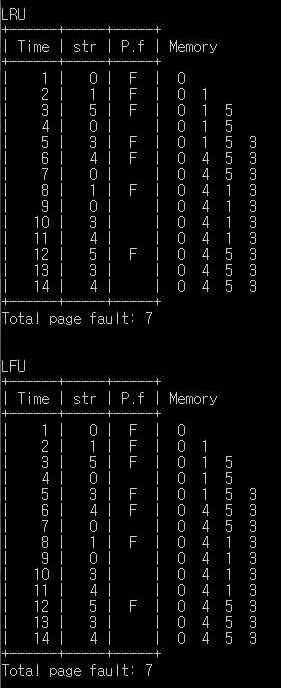
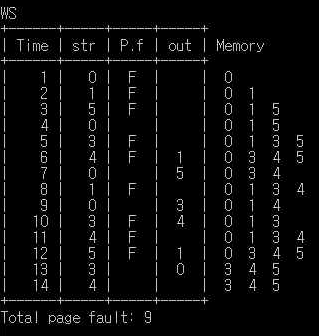
정해진 format에 맞춘 데이터를 console에서 표준 입력 함수를 통해 읽는다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| N | M | W | K |  |  |  |
| *r1* | *r2* | *r3* | *r4* | *r5* | ... | *rk* |

* 첫째 행에서...
  + N은 process가 갖는 page 개수.
  + M은 할당 page frame 개수.
  + W는 window size.
  + K는 page reference string 길이.
* 둘째 행은 page reference string.

EX)

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

출력**  **

* MIN, FIFO, LRU, LFU, WS 순서대로, 메모리 상태의 변화 과정을 표 형식으로 출력.
* 예외: WS 기법
  + 표에 **out** 칼럼 추가.
  + 메모리에 적재된 page를 번호 오름차순으로 출력한다.
* 메모리 상태 변화 과정을 출력한 뒤, 발생한 page fault의 수를 출력.

프로그램설계

**main.h**

메인 헤더에 시뮬레이션에 필요한 정적 라이브러리와 매크로 함수 명시.

**main.c**

1. 표준입력 함수를 통해 사용자로부터 입력을 받는다.
2. 시뮬레이션을 시작한다.
3. 사용한 메모리 자원을 반납하고 종료한다.

**setter.c**

set\_simulation() 함수에서 사용자의 입력을 받아, 배열에 메모리를 동적 할당하여 초기화한다. 표준 입력과 메모리 동적 할당 과정에서 생길 수 있는 예외가 각 code number와 함께 처리되어 있다.

**simulator.c**

* 고정 할당기법의 경우 fixed\_allo() 함수를 호출.
* 각 replacement 기법은 MIN\_algo(), FIFO\_algo(), LRU\_algo(), LFU\_algo() 함수에 구현하여, victim이 될 page number를 반환.
* fixed\_allo() 함수의 작동 과정:
  + page frame 초기화.
  + 현재 참조할 page를 page frame에서 찾는다.
    - frame에 존재하면, 해당 페이지를 참조하고 다음 page로 넘어간다.
  + page fault가 발생했으면, 참조할 page를 빈 page frame에 로드.
  + 현재 page frame에 빈 곳이 없으면, 정해진 replacement 기법을 적용하여 찾은 victim 자리에 로드.
* 가변 할당기법의 경우 variable\_allo() 함수를 호출.
  + page frame(working set)을 초기화.
  + 현재 시각 기준으로 working set을 구한다.
  + 현재 참조할 page를 working set에서 찾는다.
    - working set에 존재하면, 해당 페이지를 참조하고 다음 page로 넘어간다.
  + page fault가 발생했으면, page frame을 추가로 할당받아 그곳에 로드.
  + working set에서 나가는 page가 있는지 확인.

**utility.c**

동적할당한 메모리를 해제하거나, 시뮬레이션 결과를 출력하기 위한 함수가 있습이다.

테스트

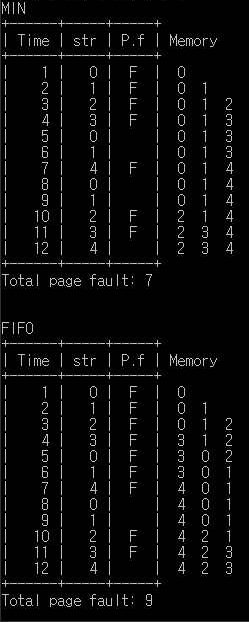
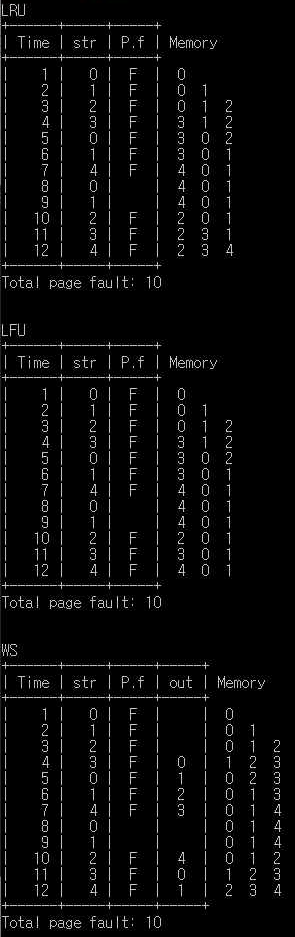
**1. MIN, LFU algorithm**의 **tie-breaking rule**

입력:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | 3 | 2 | 12 |  |  |  |  |  |  |  |  |
| 0 | 1 | 2 | 3 | 0 | 1 | 4 | 0 | 1 | 2 | 3 | 4 |

* 5개 페이지
* 3개 페이지 프레임
* window size = 2
* page reference string 길이 = 12
* page reference string = "012301401234"

출력:

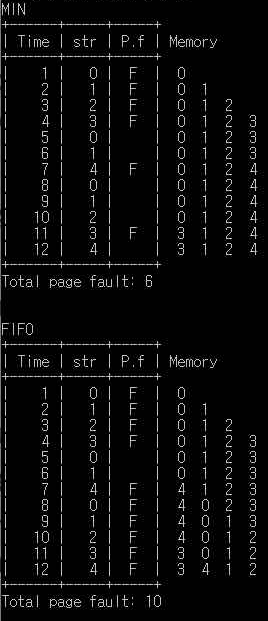
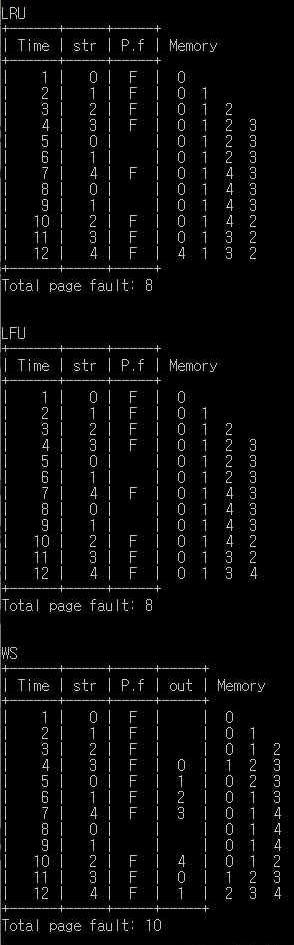
**2. FIFO anomaly**

입력:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | 4 | 2 | 12 |  |  |  |  |  |  |  |  |
| 0 | 1 | 2 | 3 | 0 | 1 | 4 | 0 | 1 | 2 | 3 | 4 |

* 5개 페이지
* **4**개 페이지 프레임
* window size = 2
* page reference string 길이 = 12
* page reference string = "012301401234"
  + page frame만 3에서 4로 추가로 하나가 더 할당된 경우.
  + FIFO anomaly. M*emory frame*이추가로할당되었으나*,* 오히려 *page fault* 발생빈도가증가

출력:

* 케이스 1과 비교했을때, page fault 발생 횟수가 1번 증가.

**3.** 정상적인케이스

입력:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 | 4 | 3 | 14 |  |  |  |  |  |  |  |  |  |  |
| 0 | 1 | 5 | 0 | 3 | 4 | 0 | 1 | 0 | 3 | 4 | 5 | 3 | 4 |

* 6개 페이지
* 4개 페이지 프레임
* window size = 3
* page reference string 길이 = 14
* page reference string = "0153401034534"

출력:

