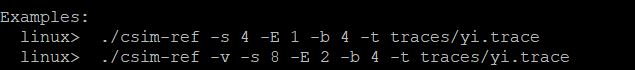
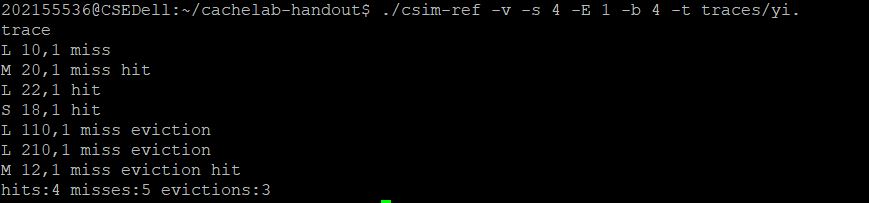
Cache lab report

202155536 김주송

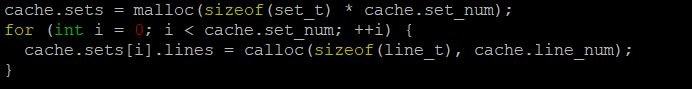
**<csim.c>**

./csim-ref 명령을 이용해

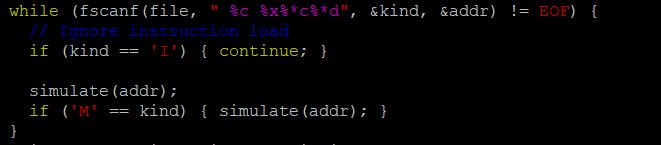
위의 명령어를 통해 s, E, b 값과 입력으로 쓰일 파일명을 입력받는 것을 알 수 있다.

그 후, Trace 폴더에 yi.trace를 확인해보면 해야할 일을 알 수 있다.

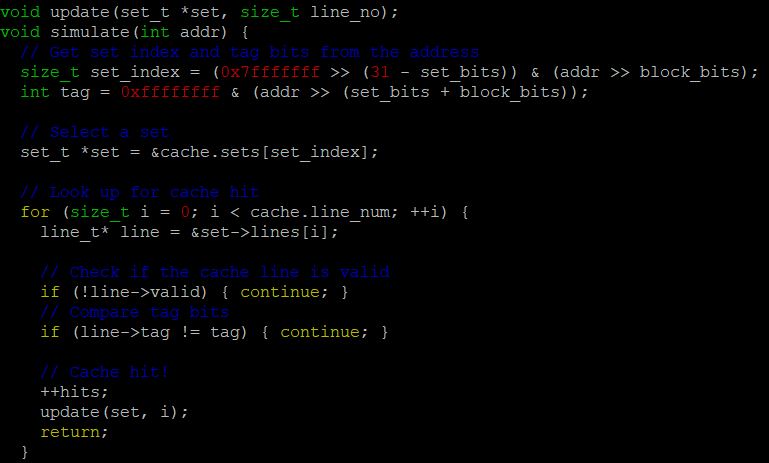
getopt 함수로 프로그램에 입력된 인자들을 파싱한다.

주어진 인자를 통해 가상의 cache 를 메모리에 생성한다.

fscanf 함수를 이용해 인자를 입력받는다.

I 입력은 무시하고, M 입력일 경우 같은 위치를 두 번 접근한다.

Eviction 알고리즘으로는 LRU를 사용하였다.

처음에 Address를 비트 연산자로 set을 고른 후, valid가 set 되어있고 같은 tag를 공유하는 cache를 찾는다.

Cache hit를 실패할 경우, 빈 cache 자리를 찾고 빈 line이 한 개도 없을 경우 가장 쓰이지 않은 오래된 cache line을 지우고 그 자리에 사용한다.

**<trans.c>**

고려해야 할 경우는 총 3가지이다.

1. 32x32 Matrix
2. 64x64 Matrix
3. 그 밖 Matrix

temporal locality을 위해 block을 사용하여 수행한다. (matrix를 cache line안에 여러부분으로 나누어 계산하는 방법 사용)

1. 32x32 Matrix

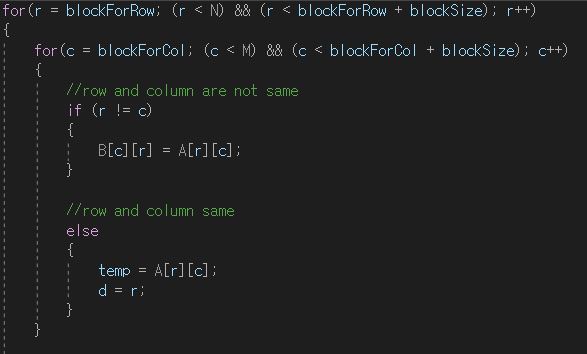
8x8 부분으로 나눠서 확인해보면, row와 column이 같을 때 대입 연산이 eviction을 일으키므로 row와 column이 같을 때와 다를 때 수행을 다르게 해준다.

1. 64x64 Matrix

Locality를 이용하여 한 번에 4개씩 받아 전치를 수행한다.

1. 그 밖의 Matrix

2개의 Loop는 column 반복에서 사용되고, 나머지 2개의 loop는 블록을 통과하는데 사용되도록 한다. 이 떄, 사이즈가 정사각형이 아닐 수 있음을 유의하며 진행한다.

blockForRow+16>N인 경우 잘못된 접근 권한을 가지게 되고, i<N, j<M도 항상 유의해줘야한다. 또한 Column과 row가 같을 때와 다를 때를 나눠 수행한다.