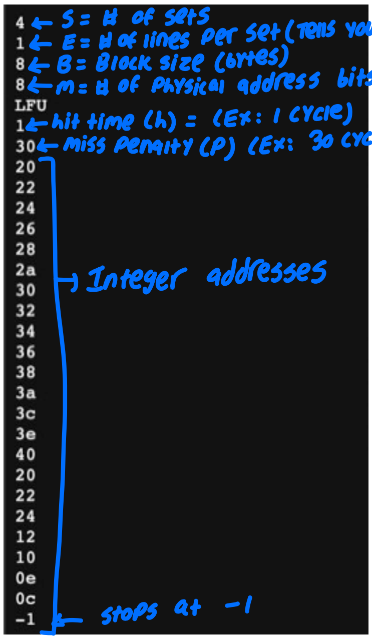
Ex input:



```
4
1
8
8
LFU
1
30
20
22
24
26
28
2a
30
32
34
36
38
3a
3c
3e
40
20
22
24
12
10
0e
0c
-1
```

4 ← S = # of sets

1 ← E = # of lines per set (Tells you if we have a $\underbrace{\text{direct mapped cache}}_{E=1}$, a $\underbrace{\text{set associative cache}}_{E>1 \text{ and } E \neq \frac{C}{B}}$, or $\underbrace{\text{a fully associative cache}}_{E = \frac{C}{B}}$).

8 ← B = Block size (bytes)

8 ← m = # of Physical address bits (# of bits in address).

1 ← hit time (h) = (Ex: 1 cycle)

30 ← miss penalty (P) (Ex: 30 cycles)

20 … 0c → Integer addresses

-1 ← stops at -1

$b = \log_2(8) = 3$ ← # of block offset bits

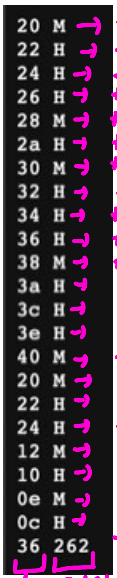
$s = \log_2(4) = 2$ ← # of set index bits

$$\text{miss rate} = \left(\frac{\text{cache misses}}{\text{\# of memory requests}}\right) \cdot 100$$

$t = 8 - 5$
$= 3$ ← # of tag bits

T	T	T	SI	SI	BO	BO	BO
0	1	2	3	4	5	6	7

Ex output:



```
20 M
22 H
24 H
26 H
28 M
2a H
30 M
32 H
34 H
36 H
38 M
3a H
3c H
3e H
40 M
20 M
22 H
24 H
12 M
10 H
0e M
0c H
36 262
```

20 M → +30 +1
22 H → +1
24 H → +1
26 H → +1
28 M → +30 +1
2a H → +1
30 M → +30 +1
32 H → +1
34 H → +1
36 H → +1
38 M → +30 +1
3a H → +1
3c H → +1
3e H → +1
40 M → +30 +1
20 M → +30 +1
22 H → +1
24 H → +1
12 M → +30 +1
10 H → +1
0e M → +30 +1
0c H → +1

262 cycles used

36 → miss rate

262 → Total cycles used

miss rate

$$= \left(\frac{\text{8 cache misses}}{\text{22 memory requests}}\right) \cdot 100 \approx 36\%$$

20 → cache miss
20, 21, 22, 23, 24, 25, 26, 27 to cache memory

22 → cache hit

24 → cache hit

26 → cache hit

28 → cache miss
28, 29, 2a, 2b, 2c, 2d, 2e, 2f to cache memory

2a → cache hit

30 → cache miss
30, 31, 32, 34, 35, 36, 37 to cache memory

32 → cache hit
34 → cache hit
36 → cache hit
38 → cache miss
38, 39, 3a, 3b, 3c, 3d, 3e, 3f to cache memory

3a → cache hit

3c → cache hit
3e → cache hit

40 → cache miss
40, 41, 42, 43, 44, 45, 46, 47 to cache memory

20 → cache miss
20, 21, 22, 23, 24, 25, 26, 27 to cache memory
22 → cache hit
24 → cache hit
12 → cache miss
10, 11, 12, 13, 14, 15, 16, 17 to cache memory

00, 01, 02, 03, 04, 05, 06, 07
08, 09, 0a, 0b, 0c, 0d, 0e, 0f
10, 11, 12, 13, 14, 15, 16, 17

t t t s s b b b

T SI BO

000000SS

Look back at Ch.2 on masks

right shift to clear block offset

(we need to extract set index and tag (2 different masks))

(when working with masks, use left shift)

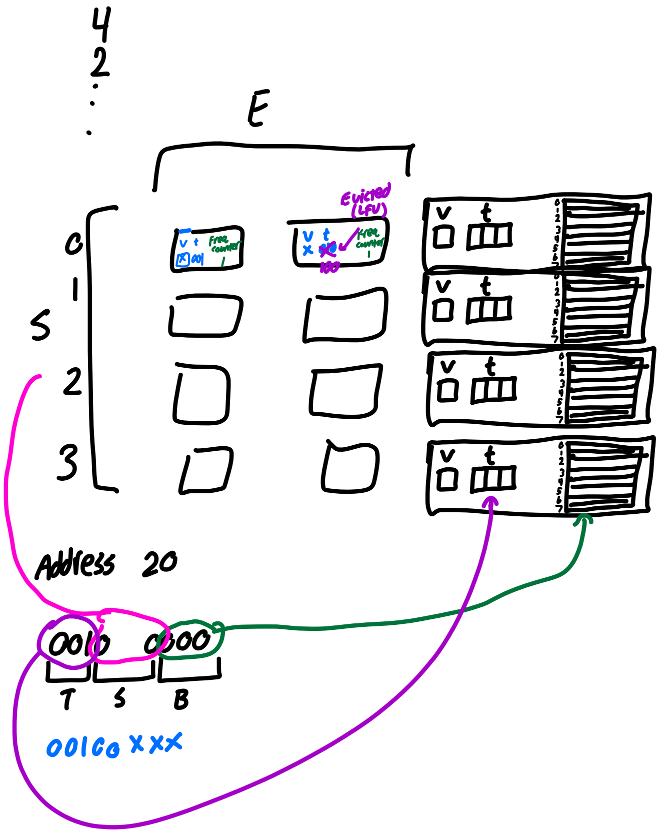
- up to 64 bits (use longs)
- we need to extract the right most bits
- For fully associative cache, there are no set index b

0010 0000

0000 0011] → 0

0010 0000

1111 1100



Address 20

0010 0000

T S B

00100xxx

Address 40

0100 0000

t s B

Address 80

1000 0000

t s B

Address 24

0010 0100

T s B

Address 40

0100 0000

t s B

- When you reload a new block, counter will reset
- Counter increases with each hit.

S=C·E

```
./yourcode < myInput > your output

diff youroutput myoutput
```