# B - Cache replacement

時間限制 2 秒

電腦存取資料時，若是每次需要資料就去記憶體尋找會耗費很多時間。因此便有快取記憶體(Cache Memory)提供又近又快速的存取，讓效能得以提升。

當有需要存取記憶體的指令出現時，Cache會將需要的主記憶體位址存進Cache Memory中以備不時之需。當一個記憶體要求被提出時，若是其已經紀錄在Cache Memory中，則稱為「命中」(Hit)；反之則稱為「未命中」(Miss)。若Cache Memory未滿且要求的記憶體位址為Miss時，則可以在Cache Memory中新增此位址紀錄，然而Cache Memory空間有限，將所有用過的位址都存下來是不可能的。因此當Cache Memory爆滿而又發生Miss時，決定誰能留下可能成為效能的關鍵。而　First in First Out (FIFO)即是其中一種常見的決定方法。

顧名思義，FIFO是將「最早使用到(被要求)的」記憶體位址置換的演算法。FIFO會把最早用到的記憶體位址從Cache Memory中「趕出去」。

如下表範例，時間序列4時，Cache Memory剛被填滿。時間序列5時，新進來的9會把在Cache中最早使用到的0取代掉。

| Time | Demand Address | Hit/Miss | Cache Contents After Demand | | |
|---|---|---|---|---|---|
| 0 | | | cache[0] | cache[1] | cache[2] |
| 1 | 0 | Miss | 0 | | |
| 2 | 8 | Miss | 0 | 8 | |
| 3 | 0 | Hit | 0 | 8 | |
| 4 | 6 | Miss | 0 | 8 | 6 |
| 5 | 9 | Miss | 8 | 6 | 9 |
| 6 | 8 | Hit | 9 | 8 | 6 |
| 7 | 9 | Hit | 9 | 8 | 6 |

給定Cache Memory 最多可以容納幾組Memory Address以及要求的Memory順序，請你利用FIFO的方法算出總共Hit了幾次。

## • 輸入格式

輸入兩行，第一行有兩個整數a, t。 a 代表最多可以容納幾組Memory Address。t 代表Memory Address個數。第二行共 t 個整數( Si，$1 \le i \le t$ )，依序代表Memory Addresses。

## • 輸出格式

輸出一行一個整數b，代表 hit 總次數。

## • 技術規格

- 本題為單筆測資
- $1 \le a \le 10^5$
- $0 \le t \le 10^5$
- $0 \le Si \le 10^5$

**範例輸入**

```
3 7
0 8 0 6 9 8 9
```

**範例輸出**

```
3
```