あそびあそばせ！

Data Structures Assignment 1
Stacks and Queues

2021.3.29

NTHU EECS

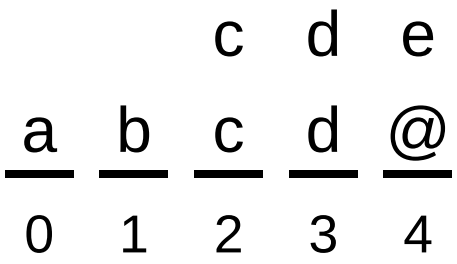
Background

- Hanako Honda, Olivia, and Kasumi Nomura are three cute junior high school students.
- One day, Hanako brought a board game she invented to play with Kasumi and Olivia.
- However, Hanako cheats for the game so that she can tease Olivia and Kasumi! You need to write a program to help those two cute junior high school students.

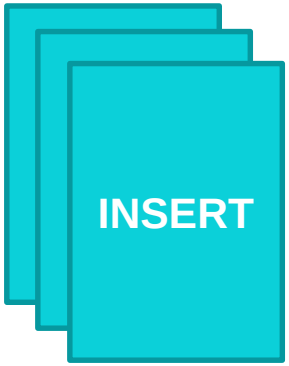
Background

Rules of the Game

- Three key components in the game:
  - The board with *L* slots
  - A stack of *D* drawing cards
  - The matching stack



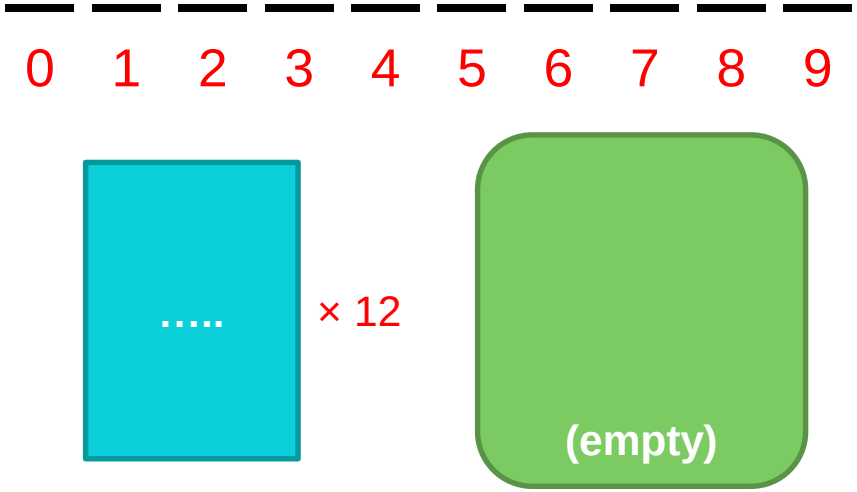
A board with slots numbered from 0 ~ L-1



A stack of D drawing cards

The matching stack

Initial Game State



Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

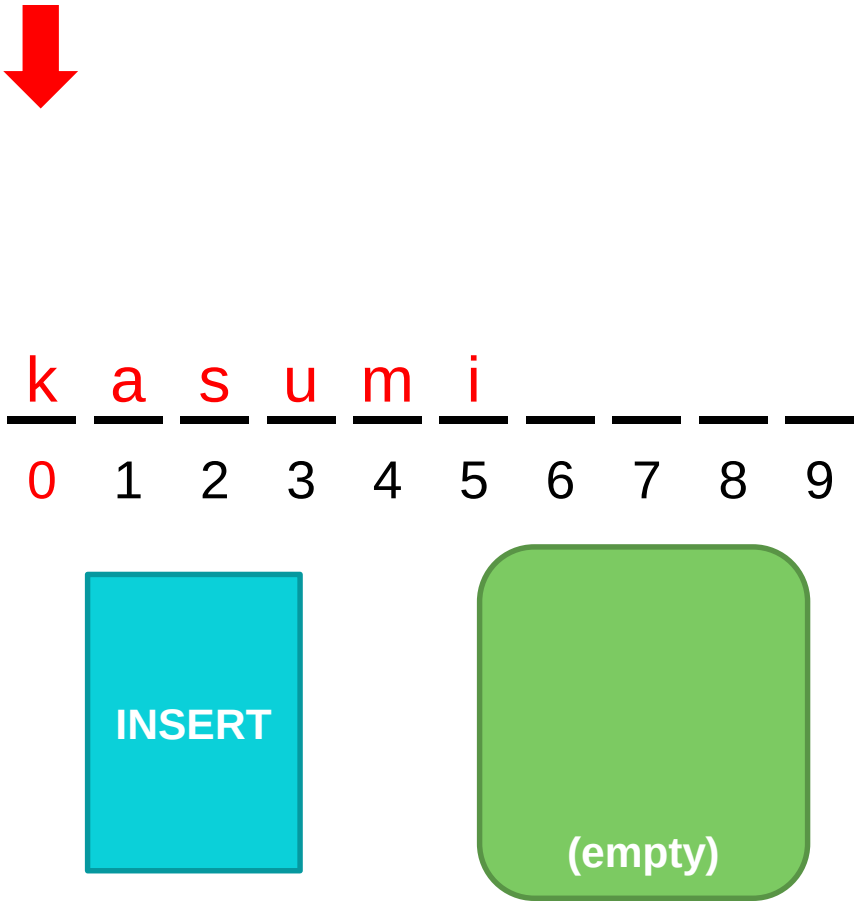
Rules of the Game

- Five types of card:
  - INSERT
  - BOTTOM_ROW
  - QUERY
  - FLUSH
  - RESET

INSERT

- Insert a string ***si*** consisting of lowercase characters starting from specific slot.
- After insertion, all empty spaces will be replaced by ‘@’ if there is at least one character above it within the same slot.
  - ‘@’ means empty obstacle.

INSERT (1/4)



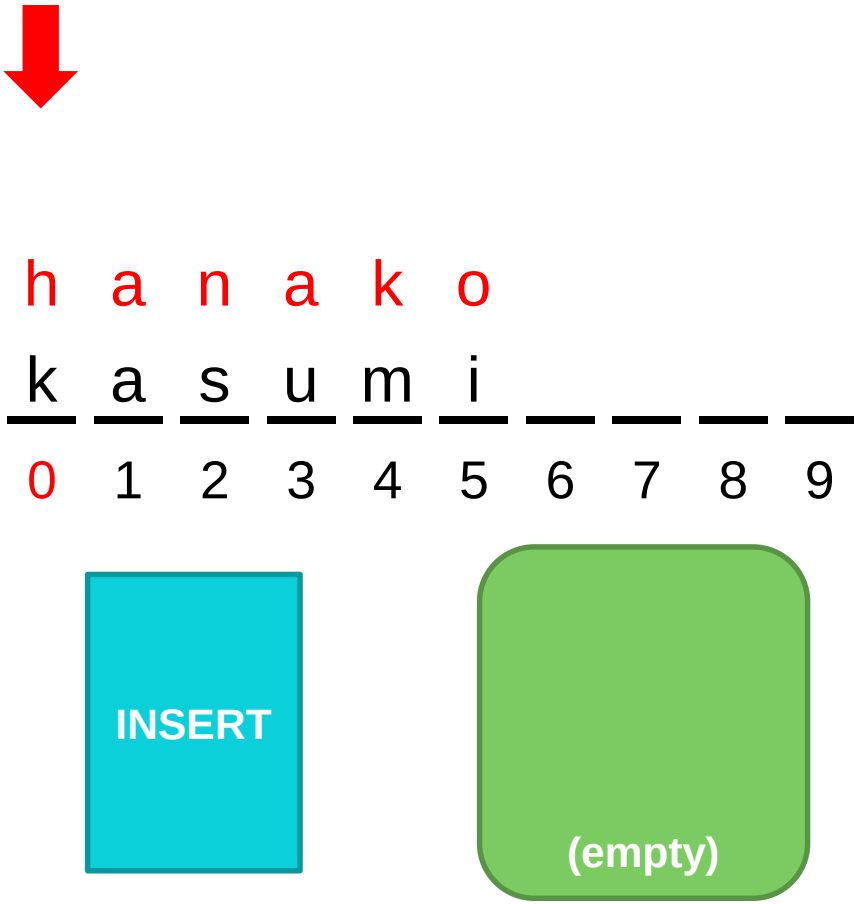
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

INSERT (2/4)



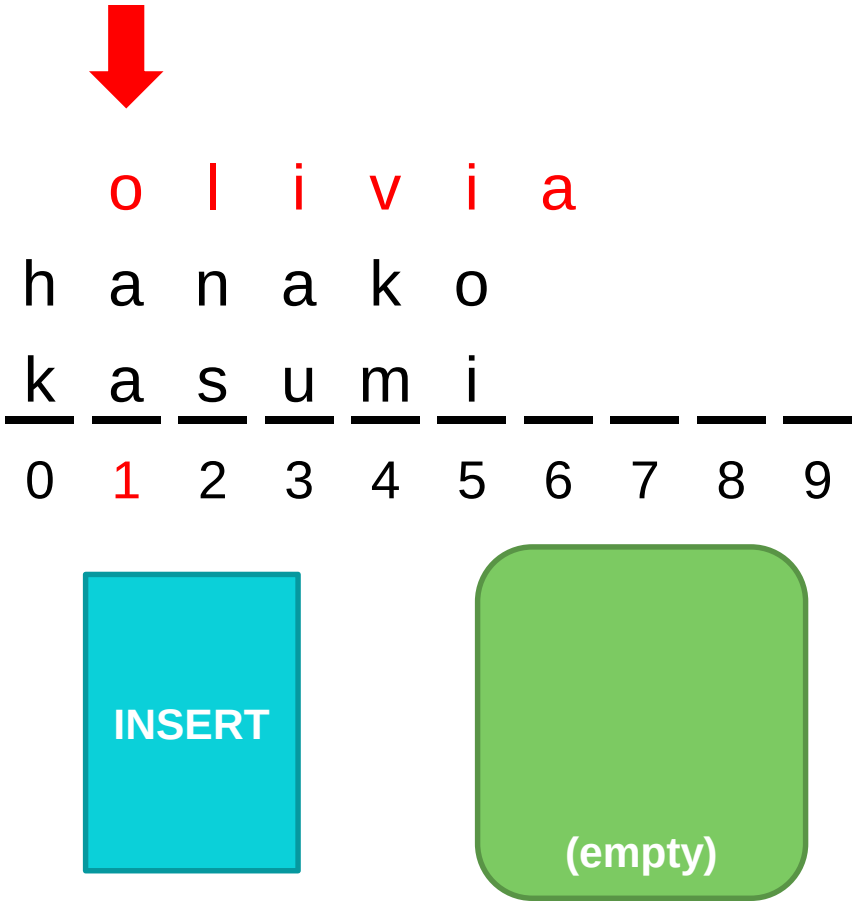
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

INSERT (3/4)



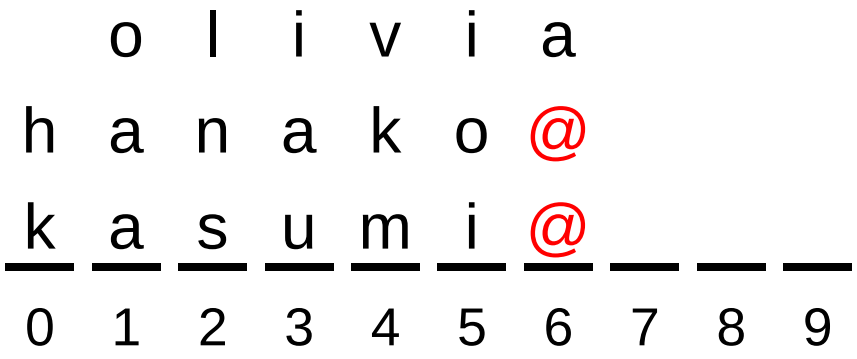
Sample Input

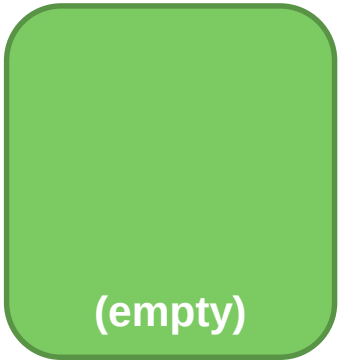
```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

INSERT (3/4)





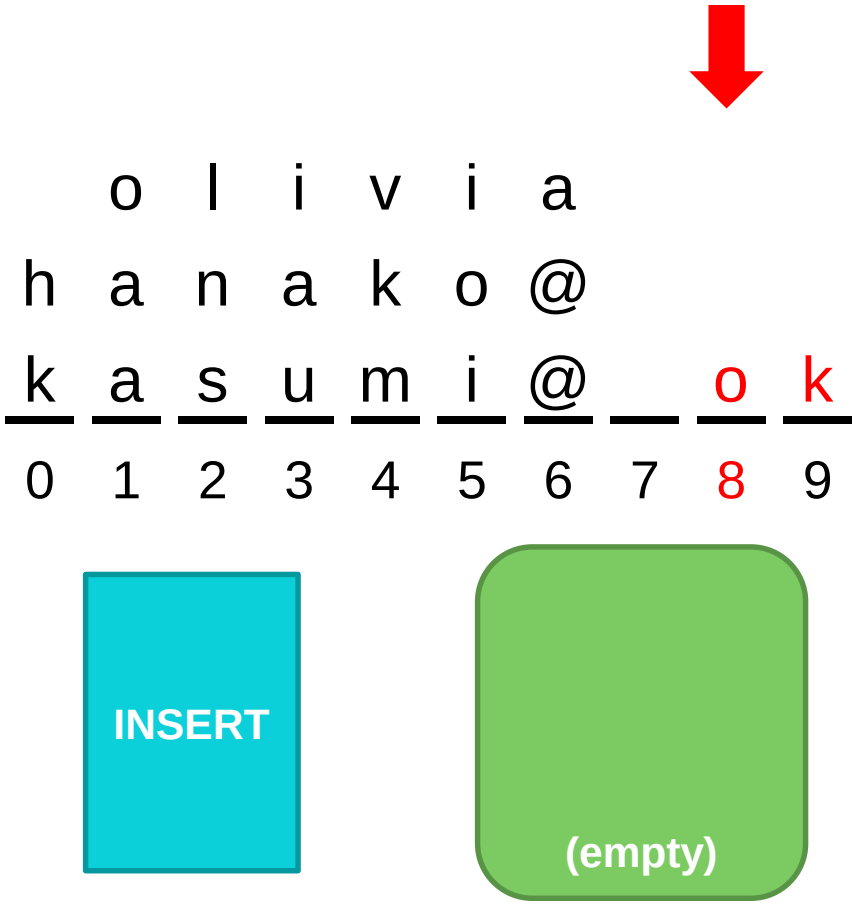
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

INSERT (4/4)



Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

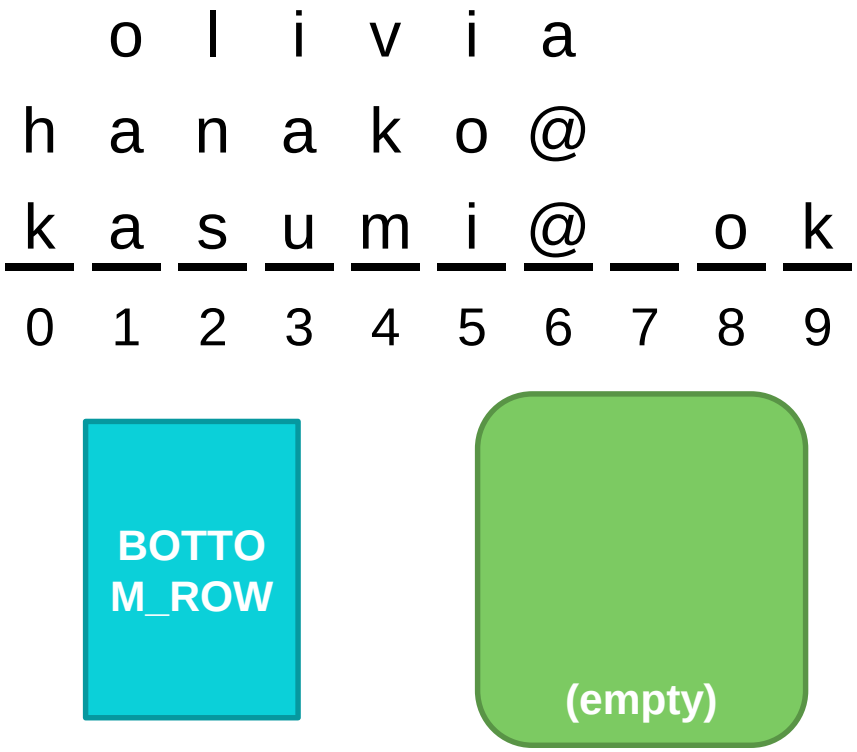
Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

BOTTOM_ROW

- Output the bottom row of the board
  - Print “BOTTOM_ROW” with ‘\n’ first. Next, print the bottom row of the board with ‘\n’.
  - Print a ‘~’ if the vertical slot contains no characters.

BOTTOM_ROW



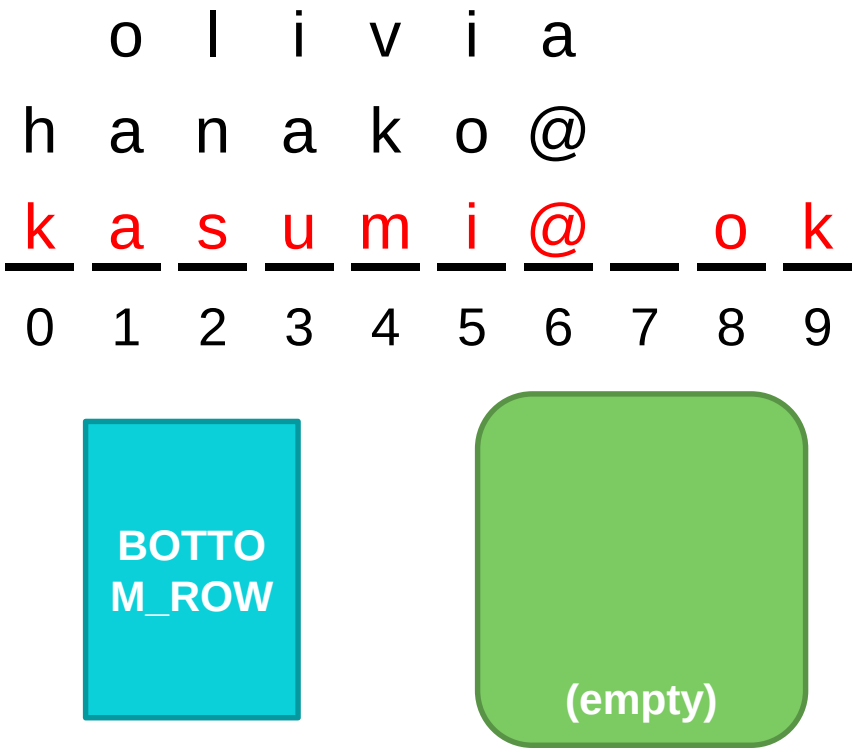
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

BOTTOM_ROW



Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

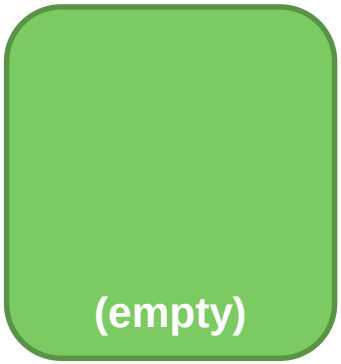
```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY

- Query a string ***sq*** to see if ***sq*** is a substring of BOTTOM_ROW string ***sb***.
  - ***sq*** consisting of lowercase characters or ‘@’.
  - If ***sq*** is a substring of ***sb***, you need to remove the string from the board and put it on top of the matching stack.

QUERY (1/5)

	o	l	i	v	i	a			
h	a	n	a	k	o	@			
k	a	s	u	m	i	@	_	o	k
0	1	2	3	4	5	6	7	8	9



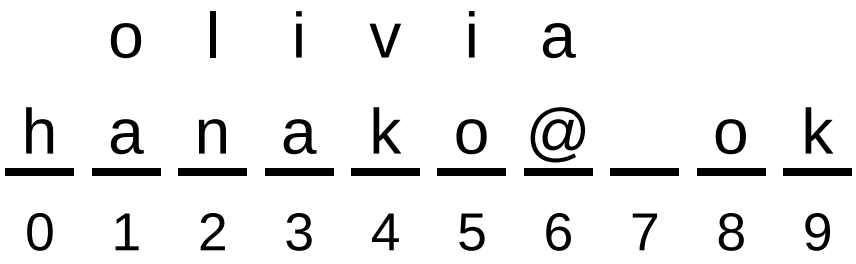
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY (1/5)



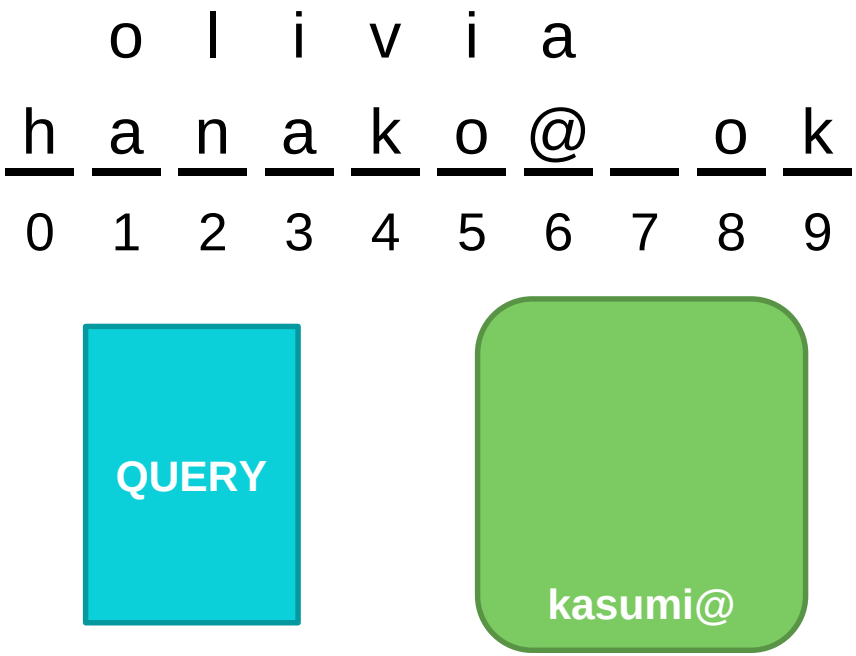
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY (2/5)



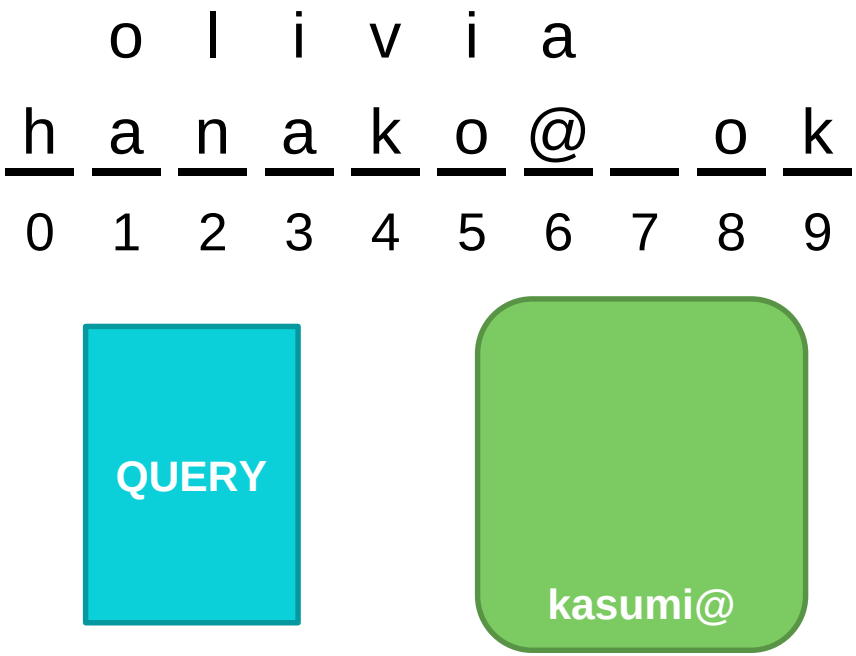
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY (3/5)



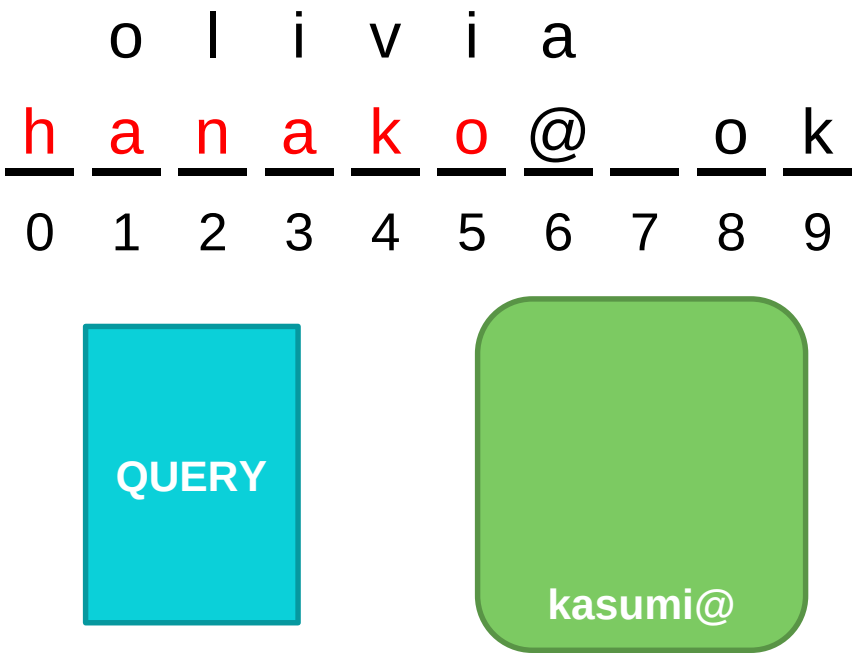
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY (4/5)



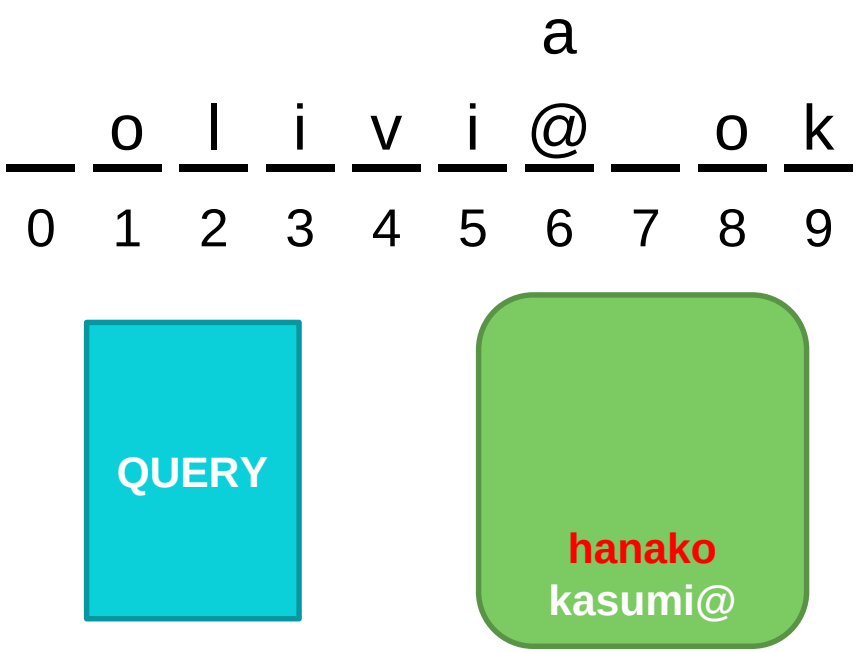
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY (4/5)



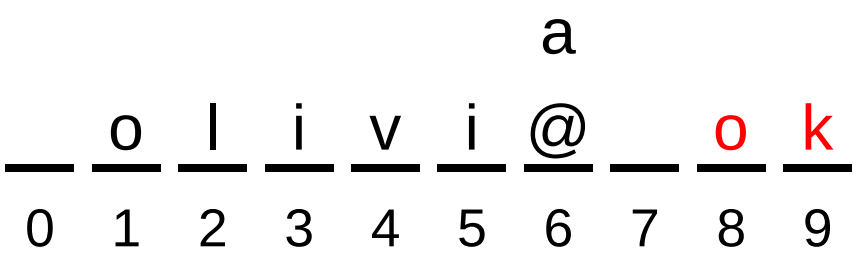
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY (5/5)



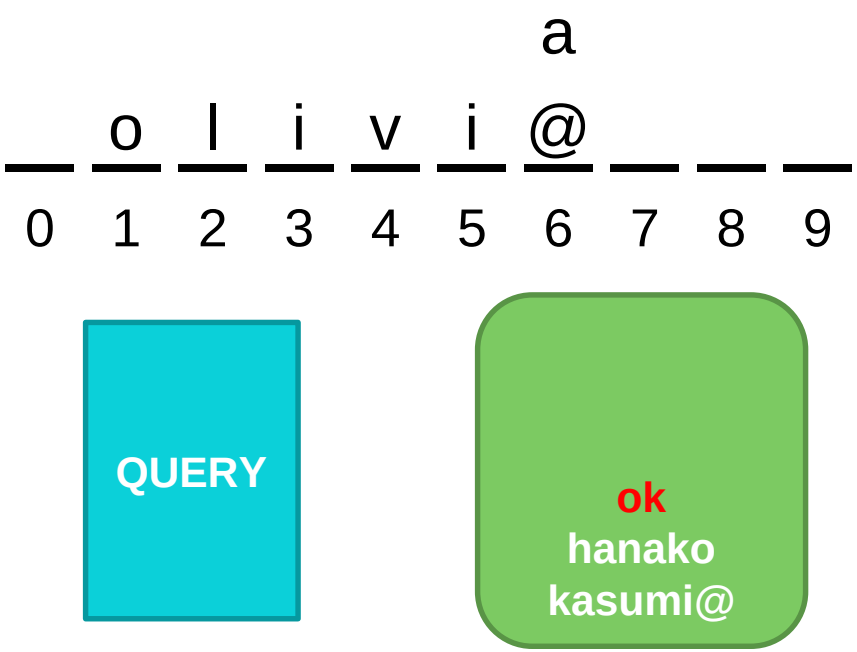
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

QUERY (5/5)



Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

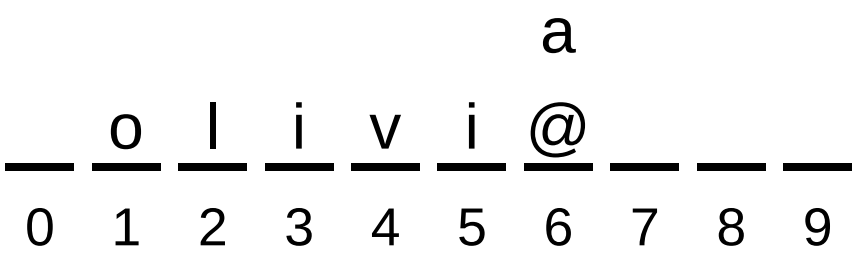
Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

FLUSH

- Print “FLUSH” with ‘\n’ first.
- Next, pop out and print all the strings in the matching stack.

FLUSH



FLUSH

ok
hanako
kasumi@

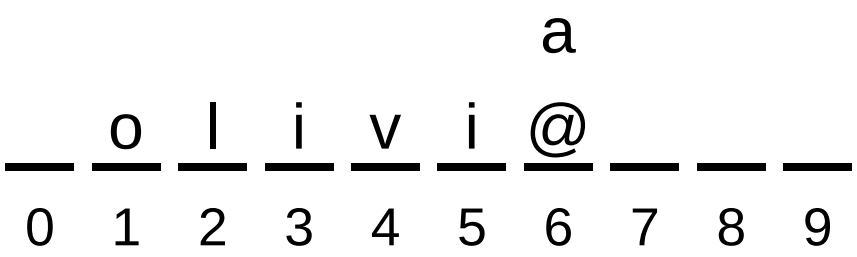
Sample Input

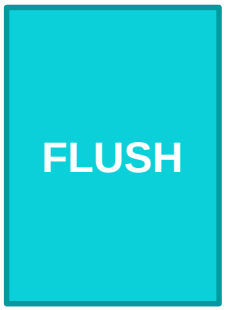
```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

FLUSH





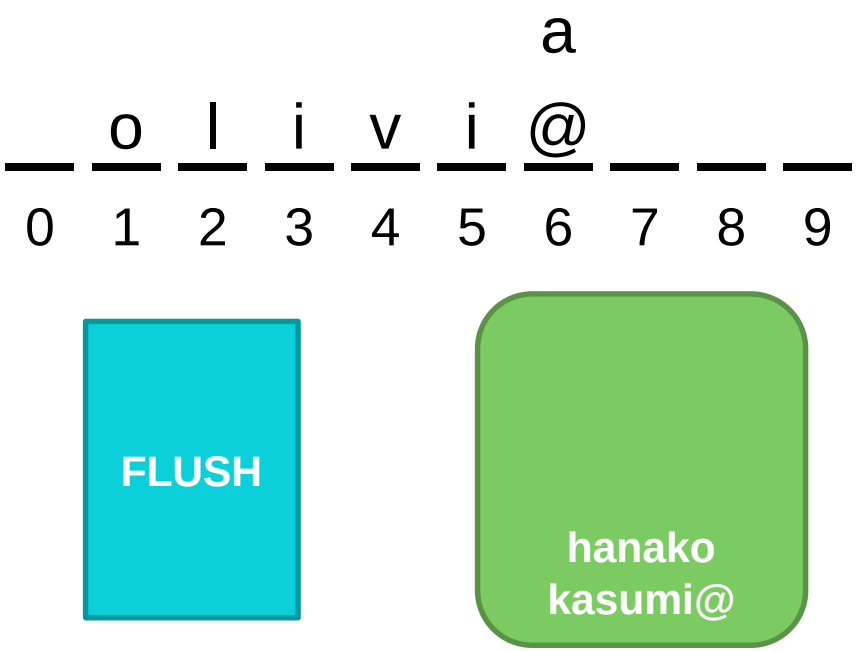
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

FLUSH



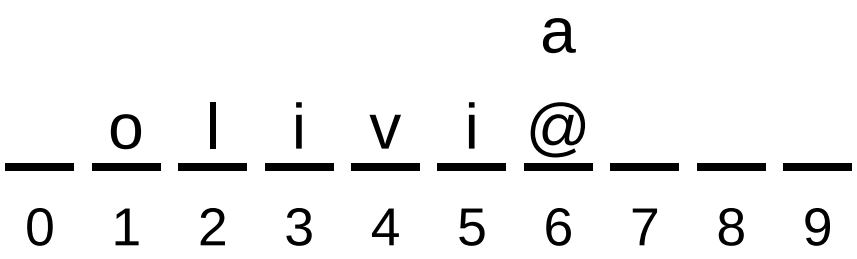
Sample Input

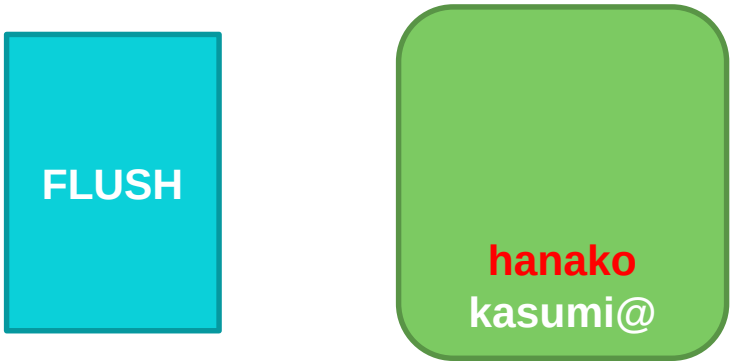
```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

FLUSH





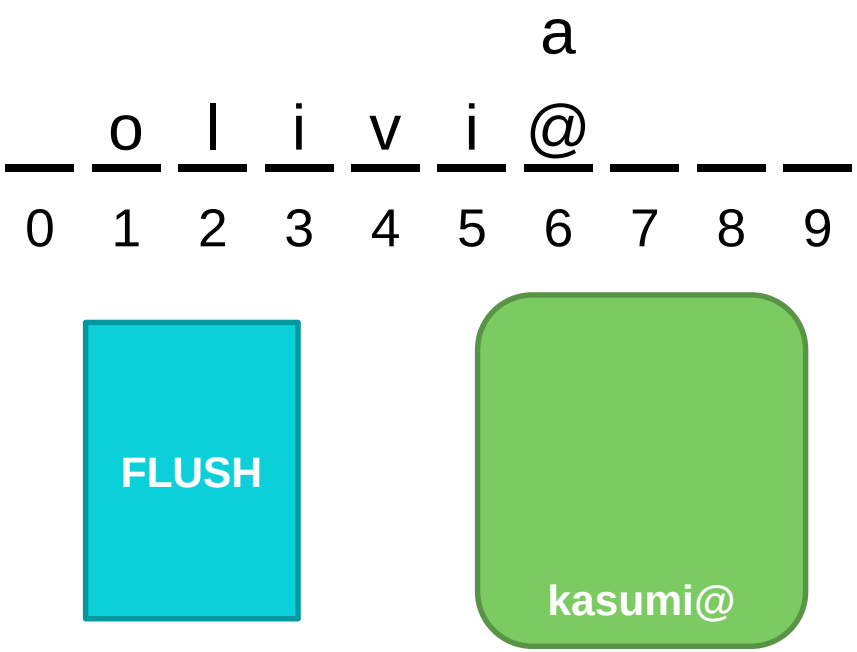
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

FLUSH



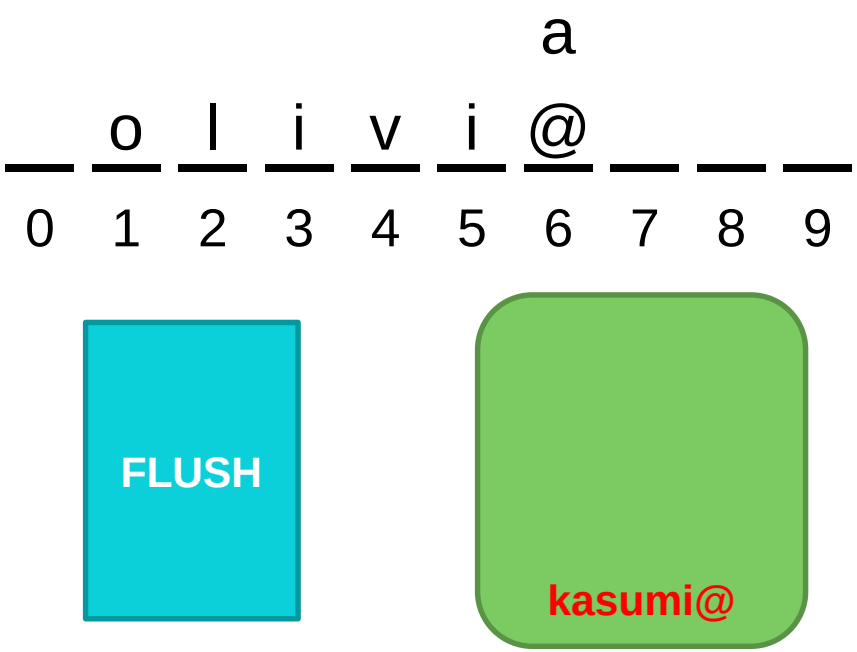
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

FLUSH



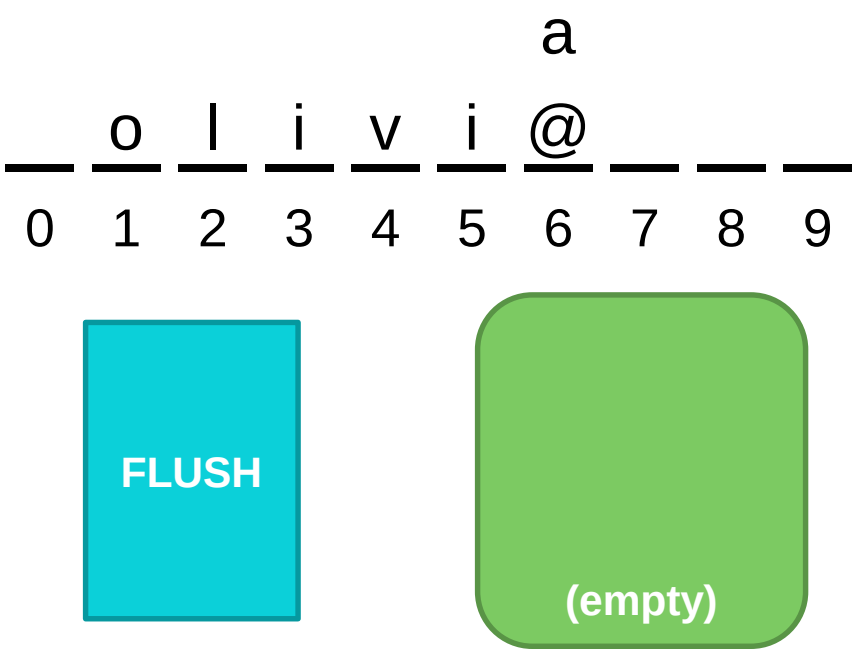
Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

FLUSH



Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

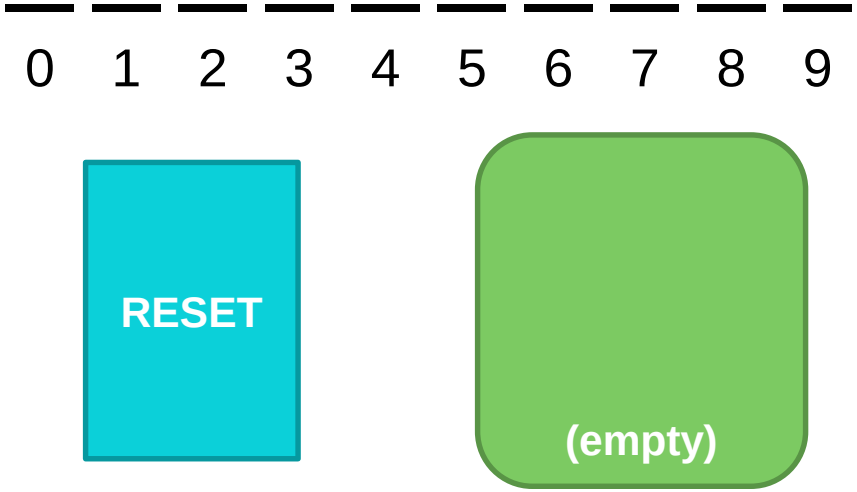
Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

RESET

- Clear out the board and the matching area.
  - Do not need to output anything.

RESET



Sample Input

```
10
12
INSERT 0 kasumi
INSERT 0 hanako
INSERT 1 olivia
INSERT 8 ok
BOTTOM_ROW
QUERY kasumi@
QUERY olivia
QUERY hanako@ok
QUERY hanako
QUERY ok
FLUSH
RESET
```

Sample Output

```
BOTTOM_ROW
kasumi@~ok
FLUSH
ok
hanako
kasumi@
```

Partial Judge

Partial Judge

- In partial judge problem, you need to solve the problem within given framework.
  - You only need to implement functions or classes defined in the header file.
- The OJ system will compile your code with the partial judge code and header.

12498 - Partial Judge Example

- Given two integer a and b, output a+b.
- Partial Judge Code:

```c
#include <stdio.h>
#include "function.h"
int main(){
    int a, b;
    scanf("%d%d",&a,&b);
    printf("%d\n",call_add(a,b));
    return 0;
}
```

- Partial Judge Header:

```c
int call_add(int a, int b);
```

12498 - Partial Judge Example

- The content below is what you need to copy and submit to OJ.

```
int call_add(int a, int b){
    return a + b;
}
```

Constraints & Hints

Constraints

- C++ containers is forbidden.
  - No credit if you use any of it.
  - <string> is allowed.
- Register the contest before deadline.
- Plagiarism is not allowed.

Hints

- Implement the code with C++ STL first.
- Brutal force substring search algorithm is fine.
- Manage memory usage carefully.
- Refer to 13144 for detailed explanations and IO constraints.

HW1 Timeline

- HW1 Registration: 3/29 11:00a.m. ~ 3/30 11a.m.
- HW1 Deadline: 4/12 12:00p.m.
- Quiz1: 4/12 18:30 ~ 20:30