

# Hashtable

## Problem description

In this problem, you need to implement a simple *hashtable* for **strings**.

*You are expected to implement a class for the hashtable.*

If the query string is in the hashtable, print “Yes”. Print “No”, otherwise. See sample I/O for clarification.

## Input/ output

Input starts with one integer *n*, where *n* is the number of entry in the hashtable. Each of the following lines contains an operation. The format of each operation is as follows:

Op-code	key
---------	-----

Op-code	Meaning
0	End of input. Ignore the key and terminate the program.
1	Insert key in the hashtable.
2	Search for the key in the hashtable and print correct ouput (Yes/ No)

See sample I/O for clarification.

## Sample input/ output

Sample input	Sample output
11 1 Jane 1 John 2 Doe 1 Doe 2 Doe 0 anything	No Yes

## Deliverables:

1. Makefile
2. Your source code

## Notes:

1. Make sure that your code runs on the CISE machines.
2. Name the executable as **“hash-table”** in your submitted Makefile
3. Follow the output format **exactly**. Do not print anything extra (e.g., prompt).
4. If **collision** occurs, feel free to implement any collision resolution of your choice (e.g., chaining/ probing).
5. Use `cin>>`, `cout<<` (`scanf`, `printf`) for I/O. Assume the input from standard input and write to standard output.
6. Assume **no whitespace** in the key. Key will be *alphanumeric*. For example, “abc01” is a valid key, “abc 01” is NOT.
7. Feel free to implement any hash function. One simple code can be as follows:

```
int hash_function(string key, int hash_table_size)
{
    int index = 0;
    for (int i=0; i<key.length(); i++)
        index += key[i];
    return index % hash_table_size;
}
```

This code will return the same hash index for both “abc” and “bac”. Can you implement a hash function that can resolve this issue (i.e., generate two different indices for these two keys)?

## Points:

Criteria	Points
Correct output	5
Code organization	3
Comments	2
Total	10