# Exercise 41: Building hash tables

Andreas Loibl

December 1, 2010

# Example hash table

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | |
| 19 | |
| 15 | |
| 20 | |
| 33 | |
| 12 | |
| 17 | |
| 10 | |

**Hash table:**

| index | keys |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | 5 |
| 6 | |
| 7 | |
| 8 | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | |
| 15 | |
| 20 | |
| 33 | |
| 12 | |
| 17 | |
| 10 | |

**Hash table:**

| index | keys |
|---|---|
| 0 | |
| 1 | 28 |
| 2 | |
| 3 | |
| 4 | |
| 5 | 5 |
| 6 | |
| 7 | |
| 8 | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | |
| 20 | |
| 33 | |
| 12 | |
| 17 | |
| 10 | |

**Hash table:**

| index | keys |
|---|---|
| 0 | |
| 1 | 28 19 |
| 2 | |
| 3 | |
| 4 | |
| 5 | 5 |
| 6 | |
| 7 | |
| 8 | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | 6 |
| 20 | |
| 33 | |
| 12 | |
| 17 | |
| 10 | |

**Hash table:**

| index | keys |
|---|---|
| 0 | |
| 1 | 28 19 |
| 2 | |
| 3 | |
| 4 | |
| 5 | 5 |
| 6 | 15 |
| 7 | |
| 8 | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | 6 |
| 20 | 2 |
| 33 | |
| 12 | |
| 17 | |
| 10 | |

**Hash table:**

| index | keys |
|---|---|
| 0 | |
| 1 | 28 19 |
| 2 | 20 |
| 3 | |
| 4 | |
| 5 | 5 |
| 6 | 15 |
| 7 | |
| 8 | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | 6 |
| 20 | 2 |
| 33 | 6 |
| 12 | |
| 17 | |
| 10 | |

**Hash table:**

| index | keys | |
|---|---|---|
| 0 | | |
| 1 | 28 | 19 |
| 2 | 20 | |
| 3 | | |
| 4 | | |
| 5 | 5 | |
| 6 | 15 | 33 |
| 7 | | |
| 8 | | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | 6 |
| 20 | 2 |
| 33 | 6 |
| 12 | 3 |
| 17 | |
| 10 | |

**Hash table:**

| index | keys | |
|---|---|---|
| 0 | | |
| 1 | 28 | 19 |
| 2 | 20 | |
| 3 | 12 | |
| 4 | | |
| 5 | 5 | |
| 6 | 15 | 33 |
| 7 | | |
| 8 | | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | 6 |
| 20 | 2 |
| 33 | 6 |
| 12 | 3 |
| 17 | 8 |
| 10 | |

**Hash table:**

| index | keys | |
|---|---|---|
| 0 | | |
| 1 | 28 | 19 |
| 2 | 20 | |
| 3 | 12 | |
| 4 | | |
| 5 | 5 | |
| 6 | 15 | 33 |
| 7 | | |
| 8 | 17 | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | 6 |
| 20 | 2 |
| 33 | 6 |
| 12 | 3 |
| 17 | 8 |
| 10 | 1 |

**Hash table:**

| index | keys | | |
|---|---|---|---|
| 0 | | | |
| 1 | 28 | 19 | 10 |
| 2 | 20 | | |
| 3 | 12 | | |
| 4 | | | |
| 5 | 5 | | |
| 6 | 15 | 33 | |
| 7 | | | |
| 8 | 17 | | |

**Hash function:**

| $k$ | $h(k) = k \mod 9$ |
|---|---|
| 5 | $5 \mod 9 = 5$ |
| 28 | $28 \mod 9 = 1$ |
| 19 | 1 |
| 15 | 6 |
| 20 | 2 |
| 33 | 6 |
| 12 | 3 |
| 17 | 8 |
| 10 | 1 |

**Hash table:**

| index | keys | | |
|---|---|---|---|
| 0 | | | |
| 1 | 28 | 19 | 10 |
| 2 | 20 | | |
| 3 | 12 | | |
| 4 | | | |
| 5 | 5 | | |
| 6 | 15 | 33 | |
| 7 | | | |
| 8 | 17 | | |

# Implementation

# in R code

```
keys <- c(5, 28, 19, 15, 20, 33, 12, 17, 10)
hash <- function(key) as.character(key %% 9)

table <- list()

for(key in keys)
  table[[hash(key)]] <- c(table[[hash(key)]], key)

for(key in sort(names(table)))
  cat(key, ":", table[[key]], "\n")
```

# Script output

```
1 : 28 19 10
2 : 20
3 : 12
5 : 5
6 : 15 33
8 : 17
```

# in C++ code

```
#include <stdio.h>
#include <stdlib.h>

#define TABLE_SIZE  9

struct node {
        int key;
        struct node *next;
};

static node *hash_table[TABLE_SIZE] = {};

int hash(int key)
{
        return key % TABLE_SIZE;
}

node *new_node()
{
        node *n = (node *) malloc(sizeof(node));
        return n;
}
```

# in C++ code

```
void insert(int key)
{
        int hashkey = hash(key);
        node *tmp = new_node();
        tmp->key = key;
        tmp->next = hash_table[hashkey];
        hash_table[hashkey] = tmp;
}

void dump_hash_table()
{
        for(int i = 0; i<TABLE_SIZE; i++)
        {
                printf("%d :", i);
                node *tmp = hash_table[i];
                while(tmp)
                {
                        printf(" %d", tmp->key);
                        tmp = tmp->next;
                }
                printf("\n");
        }
}
```

# in C++ code

```
void cleanup()
{
        for(int i = 0; i<TABLE_SIZE; i++)
        {
                while(node *tmp = hash_table[i])
                {
                        hash_table[i] = hash_table[i]->next;
                        free(tmp);
                }
        }

}

int main(int argc, char **argv)
{
        int keys[] = {5, 28, 19, 15, 20, 33, 12, 17, 10};
        for(unsigned int i = 0; i<sizeof(keys)/sizeof(int); i++)
                insert(keys[i]);
        dump_hash_table();
        cleanup();
        return(0);
}
```