**SSN College of Engineering, Kalavakkam Department of Computer Science and Engineering III Semester - CSE 'A ',’B’ & ‘C’**  
**UCS 1312 Data Structures Lab**

Academic Year: 2019-2020 Batch: 2018-2022

**Aditya Krishna P**

**185001010 || CSE A**

**Exercise 13: Implementation of hash function**

**1.**

1. **Store the following numbers in 5 buckets using any hash function (use separate chaining to avoid collision)**  35, 26, 12, 24, 43, 38, 37, 41, 22, 11, 15
2. **Search for an element in the hash table.**
3. **Delete 38 from hash table.**
4. **Display hash table after each operation.**

#include <stdio.h>

#include<stdlib.h>

#define MAX 5

struct Record

{

int data;

struct Record \*link;

};

void insert(int id, struct Record \*hash\_table[]);

int search\_element(int key, struct Record \*hash\_table[]);

void remove\_record(int key, struct Record \*hash\_table[]);

void show(struct Record \*hash\_table[]);

int hash\_function(int key);

int main()

{

struct Record \*hash\_table[MAX]; int count, key, option,id;

for(count = 0; count <= MAX - 1; count++)

{ hash\_table[count] = NULL; }

while(1)

{ printf("\n---MENU---\n");

printf("1. Insert a Record in Hash Table\n");

printf("2. Search for a Record\n");

printf("3. Delete a Record\n");

printf("4. Show Hash Table\n");

printf("5. Quit\n"); printf("Enter your option: ");

scanf("%d",&option);

switch(option)

{

case 1: {

printf("Enter the number:\t"); scanf("%d", &id);

insert(id, hash\_table);} break;

case 2: {

printf("Enter the element to search:\t"); scanf("%d", &key);

count = search\_element(key, hash\_table);

if(count == -1)

printf("Element Not Found\n");

else

printf("Element Found in Chain:\t%d\n", count); }

break;

case 3: {

printf("Enter the element to delete:\t"); scanf("%d", &key);

remove\_record(key, hash\_table); } break;

case 4: show(hash\_table); break;

case 5: exit(1);

}

}

return 0;

}

void insert(int id, struct Record \*hash\_table[])

{

int key, h;

struct Record \*temp;

key = id;

if(search\_element(key, hash\_table) != -1) {

printf("Duplicate Key\n"); return;

}

h = hash\_function(key);

temp = malloc(sizeof(struct Record));

temp->data = id;

temp->link = hash\_table[h];

hash\_table[h] = temp;

}

void show(struct Record \*hash\_table[])

{ int count;

struct Record \*ptr;

for(count = 0; count < MAX; count++)

{

printf("\n[%3d]", count);

if(hash\_table[count] != NULL)

{ ptr = hash\_table[count];

while(ptr != NULL)

{ printf(" %d ", ptr->data);

ptr=ptr->link; }

}

}

printf("\n");

}

int search\_element(int key, struct Record \*hash\_table[])

{

int h;

struct Record \*ptr;

h = hash\_function(key);

ptr = hash\_table[h];

while(ptr != NULL)

{ if(ptr->data == key)

return h;

ptr = ptr->link;

}

return -1;

}

void remove\_record(int key, struct Record \*hash\_table[])

{

int h;

struct Record \*temp, \*ptr; h = hash\_function(key);

if(hash\_table[h]==NULL) {

printf("Key %d Not Found\n", key); return;

}

if(hash\_table[h]->data == key)

{ temp = hash\_table[h];

hash\_table[h] = hash\_table[h]->link;

free(temp);

return;

}

ptr = hash\_table[h];

while(ptr->link != NULL)

{

if(ptr->link->data == key)

{

temp = ptr->link;

ptr->link = temp->link; free(temp);

return;

}

ptr = ptr->link;

}

printf("Key %d Not Found\n", key);

}

int hash\_function(int key)

{ return (key % MAX); }

**OUTPUT**

gml10:Ex13 csea10$ ./hashfn1

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 35

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 26

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 12

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 24

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 43

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 38

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 37

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 41

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 22

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 11

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the number: 15

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 4

[ 0] 15 35

[ 1] 11 41 26

[ 2] 22 37 12

[ 3] 38 43

[ 4] 24

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 2

Enter the element to search: 26

Element Found in Chain: 1

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 2

Enter the element to search: 98

Element Not Found

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 3

Enter the element to delete: 38

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 4

[ 0] 15 35

[ 1] 11 41 26

[ 2] 22 37 12

[ 3] 43

[ 4] 24

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 5

**2. Store the strings {“abcdef”, “bcdefa”, “cdefab” , “defabc” } using the following hash function.**  
**The index for a specific string will be equal to sum of ASCII values of characters multiplied by their respective order in the string after which it is modulo with 2069 (prime number)**

#include <stdio.h>

#include<stdlib.h>

#include <string.h>

#define MAX 2069

#define SIZE 20

struct Record

{

char data[SIZE];

struct Record \*link;

};

void insert(char id[], struct Record \*hash\_table[]);

int search\_element(char key[], struct Record \*hash\_table[]);

void remove\_record(char key[], struct Record \*hash\_table[]);

void show(struct Record \*hash\_table[]);

int hash\_function(char key[]);

int main()

{

struct Record \*hash\_table[MAX]; int count,option;

char id[MAX], key[MAX];

for(count = 0; count <= MAX - 1; count++)

{ hash\_table[count] = NULL; }

while(1)

{ printf("\n---MENU---\n");

printf("1. Insert a Record in Hash Table\n");

printf("2. Search for a Record\n");

printf("3. Delete a Record\n");

printf("4. Show Hash Table\n");

printf("5. Quit\n"); printf("Enter your option: ");

scanf("%d",&option);

switch(option)

{

case 1: {

printf("Enter the string:\t"); scanf("%s", id);

insert(id, hash\_table);} break;

case 2: {

printf("Enter the element to search:\t"); scanf("%s", key);

count = search\_element(key, hash\_table);

if(count == -1)

printf("Element Not Found\n");

else

printf("Element Found in Chain:\t%d\n", count); }

break;

case 3: {

printf("Enter the element to delete:\t");

scanf("%s", key);

remove\_record(key, hash\_table); } break;

case 4: show(hash\_table); break;

case 5: exit(1);

}

}

return 0;

}

void insert(char id[], struct Record \*hash\_table[])

{

char key[SIZE];

int h;

struct Record \*temp;

strcpy(key, id);

if(search\_element(key, hash\_table) != -1) {

printf("Duplicate Key\n"); return;

}

h = hash\_function(key);

temp = malloc(sizeof(struct Record));

strcpy(temp->data,id);

temp->link = hash\_table[h];

hash\_table[h] = temp;

}

void show(struct Record \*hash\_table[])

{ int count;

struct Record \*ptr;

for(count = 0; count < MAX; count++)

{

if(hash\_table[count] != NULL)

{ printf("\n[%3d]", count);

ptr = hash\_table[count];

while(ptr != NULL)

{ printf(" %s ", ptr->data);

ptr=ptr->link; }

}

}

printf("\n");

}

int search\_element(char key[], struct Record \*hash\_table[])

{

int h;

struct Record \*ptr;

h = hash\_function(key);

ptr = hash\_table[h];

while(ptr != NULL)

{ if(strcmp(ptr->data,key)==0)

return h;

ptr = ptr->link;

}

return -1;

}

void remove\_record(char key[], struct Record \*hash\_table[])

{

int h;

struct Record \*temp, \*ptr;

h = hash\_function(key);

if(hash\_table[h]==NULL) {

printf("Key %s Not Found\n", key); return;

}

if( strcmp(hash\_table[h]->data, key)==0)

{ temp = hash\_table[h];

hash\_table[h] = hash\_table[h]->link;

free(temp);

return;

}

ptr = hash\_table[h];

while(ptr->link != NULL)

{

if(ptr->link->data == key)

{

temp = ptr->link;

ptr->link = temp->link;

free(temp);

return;

}

ptr = ptr->link;

}

printf("Key %s Not Found\n", key);

}

int hash\_function(char key[])

{ int l=strlen(key);

int x=0;

int a;

for (int i=1; i<=l; i++) {

a=key[i-1];

x+=a\*i;

}

return x%2069;

}

**OUTPUT**

ml10:Ex13 csea10$ ./hashfn2

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the string: abcdef

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 4

[ 38] abcdef

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the string: bcdefa

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the string: cdefab

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 1

Enter the string: defabc

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 4

[ 11] defabc

[ 14] cdefab

[ 23] bcdefa

[ 38] abcdef

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 2

Enter the element to search: abcdef

Element Found in Chain: 38

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 3

Enter the element to delete: axyz

Key axyz Not Found

---MENU---

1. Insert a Record in Hash Table

2. Search for a Record

3. Delete a Record

4. Show Hash Table

5. Quit

Enter your option: 5 