#include <stdio.h>

#include<stdlib.h>

#define MAX 20

struct Employee

{

int employee\_id;

char employee\_name[45];

int employee\_age;

};

struct Record

{

struct Employee data;

struct Record \*link;

};

void insert(struct Employee employee\_record, 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[10];

struct Employee employee\_record;

int i, key, option;

for(i = 0; i <10; i++)

{

hash\_table[i] = NULL;

}

while(1)

{

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\n");

scanf("%d",&option);

switch(option)

{

case 1:

printf("Enter the Employee Details\n");

printf("Employee ID:\t");

scanf("%d", &employee\_record.employee\_id);

printf("Employee Name:\t");

scanf("%s", employee\_record.employee\_name);

printf("Employee Age:\t");

scanf("%d", &employee\_record.employee\_age);

insert(employee\_record, hash\_table);

break;

case 2:

printf("Enter the employ ID to search:\t");

scanf("%d", &key);

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

if(i == -1)

{

printf("Element Not Found\n");

}

else

{

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

}

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(struct Employee employee\_record, struct Record \*hash\_table[])

{

int key, h;

struct Record \*temp;

key = employee\_record.employee\_id;

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

{

printf("Duplicate Key\n");

return;

}

h = hash\_function(key);

temp = malloc(sizeof(struct Record));

temp->data = employee\_record;

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

hash\_table[h] = temp;

}

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

{

int i;

struct Record \*ptr;

for(i= 0; i< 10; i++)

{

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

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

{

ptr = hash\_table[i];

while(ptr != NULL)

{

printf("%d %s %d\t", ptr->data.employee\_id, ptr->data.employee\_name, ptr->data.employee\_age);

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.employee\_id == 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.employee\_id == key)

{

temp = hash\_table[h];

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

free(temp);

printf("\n element deleted!");

return;

}

ptr = hash\_table[h];

while(ptr->link != NULL)

{

if(ptr->link->data.employee\_id == key)

{

temp = ptr->link;

ptr->link = temp->link;

free(temp);

printf("\n element deleted!");

return;

}

ptr = ptr->link;

}

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

}

int hash\_function(int key)

{

return (key % 10);

}