#include <iostream>

#include <cstring>

#define MAX 18

#define MOB 11

using namespace std;

class student

{

char name[50];

char number[MOB];

public:

student()

{

name[0] = number[0] = '-';

name[1] = number[1] = '\0';

}

void getData();

void changeName()

{

cin.ignore();

cin.getline(name, MAX);

}

void display()

{

cout << name << "\t" << number << "\t";

}

char\* getNumber()

{

return number;

}

};

void student::getData()

{

int flag, length;

cout << "\nEnter the name of the student ";

cin.ignore();

cin.getline(name, 50);

do

{

flag = 0;

cout << "\nEnter the mobile number for the student ";

cin >> number;

if (strlen(number)!= 10)

cout << "\nPlease enter a valid mobile number ";

}while (strlen(number)!= 10);

};

class hashTable

{

student data[MAX];

int chain[MAX];

int count;

public:

hashTable()

{

count = 0;

for (int i = 0; i < MAX; i++)

chain[i] = -1;

}

void insert();

void insertR();

int hash(char[MOB]);

void displayAll();

int search(char[MOB]);

void modify();

void remove();

int isEmpty();

int isFull();

};

int hashTable::hash(char key[MOB])

{

int a,b;

a=key[4]-48;

b=key[5]-48;

return a+b;

}

int hashTable::isEmpty()

{

if (count == 0)

return 1;

return 0;

}

int hashTable::isFull()

{

if (count == MAX)

return 1;

return 0;

}

void hashTable::insert() // Check if table is full before calling

{

int index, oldIndex;

student newData;

newData.getData();

index = hash(newData.getNumber());

if(strcmp(data[index].getNumber(), "-\0") != 0)

{

while (chain[index] != -1)

index = chain[index];

oldIndex = index;

while (strcmp(data[index].getNumber(), "-\0") != 0)

index = (index + 1) % MAX;

chain[oldIndex] = index;

}

data[index] = newData;

count++;

}

void hashTable::insertR() // Check if table is full before calling

{

int index, oldIndex, tempChain, i;

int flag;

student newData, tempData;

newData.getData();

index = hash(newData.getNumber());

if (strcmp(data[index].getNumber(), "-\0") != 0)

{

if (hash(data[index].getNumber()) == index)

{

while (chain[index] != -1)

index = chain[index];

oldIndex = index;

while (strcmp(data[index].getNumber(), "-\0") != 0)

index = (index + 1) % MAX;

chain[oldIndex] = index;

}

else

{

tempData = data[index];

tempChain = chain[index];

data[index] = newData;

chain[index] = -1;

oldIndex = index;

while (strcmp(data[index].getNumber(), "-\0") != 0)

index = (index + 1) % MAX;

chain[index] = tempChain;

for (i = 0; i < count+1; i++)

if (chain[i] == oldIndex)

{

flag = 1;

break;

}

if (flag == 1)

oldIndex = i;

chain[oldIndex] = index;

newData = tempData;

}

}

data[index] = newData;

count++;

}

void hashTable::displayAll()

{

cout << "Name\tMobile Number\t\tChain" << endl;

for (int i = 0; i < MAX; i++)

{

data[i].display();

cout << "\t "<<chain[i] << endl;

}

}

int hashTable::search(char number[MOB])

{

int index = hash(number);

while (true)

{

if (strcmp(number, data[index].getNumber()) == 0)

{

cout << "Record found." << endl;

cout << "Name\tMobile Number" << endl;

data[index].display();

cout << endl;

return index;

}

else if (chain[index] != -1)

index = chain[index];

else

break;

}

cout << "Record not found." << endl;

return -1;

}

void hashTable::modify()

{

char number[MOB];

int index;

cout << "Enter the mobile number of the student who's record you wish to modify - ";

cin >> number;

index = search(number);

if (index == -1)

return;

cout << "\nEnter the modified record (name only)- ";

data[index].changeName();

}

int main()

{

hashTable h;

int ch,ch1,res;

char temp[15];

do

{

cout<<"\n 1.Insert \n2.Display \n3.Search \n4.Modify";

cin>>ch;

switch(ch)

{

case 1:

if(!h.isFull())

{

cout<<"If collision occours resolve it\n 1.Without replacement \n2.With replacement";

cin>>ch1;

if(ch1==1)

h.insert();

else

h.insertR();

}

else

cout<<"\n Table is full";

break;

case 2:

h.displayAll();

break;

case 3:

cout<<"\n Enter the mobile no to be searched";

cin>>temp;

res=h.search(temp);

if(res!=-1)

cout<<"\n Entered mobile number found ";

break;

case 4:

h.modify();

}

}while(ch!=0);

}

1.Insert

2.Display

3.Search

4.Modify1

If collision occours resolve it

1.Without replacement

2.With replacement1

Enter the name of the student adi

Enter the mobile number for the student 9175593941

1.Insert

2.Display

3.Search

4.Modify1

If collision occours resolve it

1.Without replacement

2.With replacement1

Enter the name of the student adi2

Enter the mobile number for the student 9175593941

1.Insert

2.Display

3.Search

4.Modify2

Name Mobile Number Chain

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

adi 9175593941 15

adi2 9175593941 -1

- - -1

- - -1

1.Insert

2.Display

3.Search

4.Modify1

If collision occours resolve it

1.Without replacement

2.With replacement2

Enter the name of the student adi3

Enter the mobile number for the student 8796229881

1.Insert

2.Display

3.Search

4.Modify1

If collision occours resolve it

1.Without replacement

2.With replacement2

Enter the name of the student adi5

Enter the mobile number for the student 8796229881

1.Insert

2.Display

3.Search

4.Modify2

Name Mobile Number Chain

- - -1

- - -1

- - -1

- - -1

adi3 8796229881 5

adi5 8796229881 -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

- - -1

adi 9175593941 15

adi2 9175593941 -1

- - -1

- - -1

1.Insert

2.Display

3.Search

4.Modify

1.Insert

2.Display

3.Search

4.Modify3

Enter the mobile no to be searched9175593941

Record found.

Name Mobile Number

adi 9175593941

Entered mobile number found

1.Insert

2.Display

3.Search

4.Modify