#include <iostream>

#include <iomanip>

using namespace std;

#define SIZE 10

class Record {

public:

private:

string name;

int phone;

string flag; //EMPTY, OCCUPIED

friend class PhoneBook;

};

class PhoneBook {

public:

PhoneBook() {

capacity = 0;

}

void insertRecord(Record table[]);

void displayAll(Record table[]);

void searchRecord(Record table[]);

int getHashCode(int key);

private:

int capacity;

};

void PhoneBook::insertRecord(Record table[]) {

if (capacity == SIZE) {

cout << "Phonebook is full...insertion not possible\n";

} else {

Record r;

cout << "Enter name: \n";

cin>>r.name;

cout << "Enter phone number: \n";

cin >> r.phone;

int hashIndex = getHashCode(r.phone);

while (table[hashIndex].flag == "OCCUPIED") {

hashIndex = (++hashIndex % SIZE);//% to make sure array index not exceed SIZE

}

table[hashIndex] = r;

table[hashIndex].flag = "OCCUPIED";

capacity++;

}

}

void PhoneBook::displayAll(Record table[]) {

for (int i = 0; i < SIZE; ++i) {

if (table[i].flag == "OCCUPIED")

{

cout << setw(20) << left << i <<"= " << table[i].name << " => "

<< table[i].phone << endl;

}

}

}

void PhoneBook::searchRecord(Record table[]) {

int temp;

int count = 0;

cout << "\nEnter name to find its number ...\n";

cin >> temp;

int hashIndex = getHashCode(temp);

while (table[hashIndex].flag == "OCCUPIED") {

if (table[hashIndex].phone == temp) {

cout << "Name is present with phone number: "

<< table[hashIndex % SIZE].phone << endl;

return;

} else if (count == SIZE) {

cout << "Name is not present" << endl;

return;

} else {

hashIndex = (++hashIndex % SIZE);

++count;

}

}//while

cout << "Name is not present" << endl;//3 names with SIZE=20 and name not present

}

int PhoneBook::getHashCode(int key) {

int sum = 1,rem=0,n1;

while (key != 0)

{

rem = key % 10;

sum = sum \* rem;

key = key / 10;

}

n1=sum%SIZE;

return n1;

}

int main() {

int choice;

Record table[SIZE];

PhoneBook d;

do {

cout << "\n1.Insert Record\n2.Display\n3.Search\n4.Exit\n";

cout << "Enter option \n";

cin >> choice;

switch (choice) {

case 1:

d.insertRecord(table);

break;

case 2:

d.displayAll(table);

break;

case 3:

d.searchRecord(table);

break;

case 4:

break;

default:

cout << "\n Invalid Entry \n";

break;

}

} while (choice != 4);

}