****

Consider telephone book database of N clients. Make use of a hash table implementation to quickly look up client‘s telephone number. Make use of two collision handling techniques and compare them using number of comparisons required to find a set of telephone numbers

****

```
import java.util.Scanner;

class HashTable {

    public int index;

    public long mobile;

}



class Hashing {

    HashTable[] h = new HashTable[10];



    public Hashing() {

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

            h[i] = new HashTable();

            h[i].index = i;

            h[i].mobile = -1;

        }

    }



    public void display() {

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

            System.out.println(h[i].index + "  " + h[i].mobile);

        }
```

```
}


public void insert(int probchoice) {
    long key;
    int position;
    Scanner input = new Scanner(System.in);
    System.out.println("\nEnter mobile number to insert in to hash table : ");
    key = input.nextLong();
    position = (int) (key % 10);
    System.out.println("\nPosition = " + position);


    if(h[position].mobile == -1) {
        h[position].mobile = key;
    } else if(probchoice == 1) { // Linear Probing collision.
        int temp_position;
        temp_position = LinearProbing(position);
        h[temp_position].mobile = key;
    } else if(probchoice == 2) { // Quadratic Probing for collision.
        int temp_position;
        temp_position = QuadraticProbing(key);
        h[temp_position].mobile = key;
    }
}


public void search() {
    long key;

    int position;

    Scanner input = new Scanner(System.in);

    System.out.println("\nEnter mobile number to search in the hash table : ");

    key = input.nextLong();

    position = (int) (key % 10);


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

        if(h[i].mobile == key) {

            System.out.println("\nGiven mobile number is found in the hash table ");

            break;

        }

    }

    if(i == 10) {

        System.out.println("\nGiven mobile number is not found in the hash table ");

    }

}


public int LinearProbing(int collision_position) {

    for(int i = collision_position; i < 10; i++) {

        if(h[i].mobile == -1) {

            return i;

        }

        if(i == 9) {

            i = -1;
            }
        }
        return -1;
    }

    public int QuadraticProbing(long key) {
        int a;
        for(int j = 0; j < 10; j++) {
            a = (int) ((key + (j * j)) % 10);
            if(h[a].mobile == -1) {
                return a;
            }
        }
        return -1;
    }
}

class Main {
    public static void main(String[] args) {
        Hashing H = new Hashing();
        int ch;
        Scanner input = new Scanner(System.in);

        do {
            System.out.println("\n Menu");
System.out.println(" 1. insert");

System.out.println(" 2. display");

System.out.println(" 3. search");

System.out.println(" 4. exit");

System.out.println(" Enter your choice : ");

ch = input.nextInt();


switch(ch) {

   case 1: //insert

      int probchoice;

      System.out.println("Enter \n1 for LinearProbing and \n2 for Quadratic probing");

      probchoice = input.nextInt();

      H.insert(probchoice);

      break;

   case 2: //display

      H.display();

      break;

   case 3: //search

      H.search();

      break;

    case 4:

        System.exit(4);


   default:

      System.out.println("\nWrong choice :");
```

```
                break;
            }
        } while(ch != 5);


        H.display();
    }
}
```