Câu 1: Hai thuộc tính quan trọng của hàm băm là:

* Hạn chế xung đột (Không trùng địa chỉ, phân bổ đều)
* Tính toán đơn giản

Câu 2: Draw the 11-entry hash that results from using the hash function

h(i) = (2i+5) mod 11 to hash keys 12, 44, 13, 88, 23, 94, 11, 39, 20, 16, 5.

Xử lí:  
h(12) = (2\*12 + 5) mod 11 = 7

h(44) = 5, h(88) = 5, h(23) = 5, h(11) = 5;

h(13) = 9

h(94) = 6, h(39) = 6

h(20) = 1

h(16) = 4, h(5) = 4

1. Xử lí xung đột theo cách nối chuỗi:

|  |  |
| --- | --- |
| 1 | 20 |
| 2 |  |
| 3 |  |
| 4 | 16, 5 |
| 5 | 44, 88, 23, 11 |
| 6 | 94, 39 |
| 7 | 12 |
| 8 |  |
| 9 | 13 |
| 10 |  |
| 11 |  |

1. Xử lí xung đột theo duyệt tuyến tính:

Arr[11] = {12, 44, 13, 88, 23, 94, 11, 39, 20, 16, 5}

|  |  |
| --- | --- |
| 0 | 11(h=5) |
| 1 | 39(h=6) |
| 2 | 20(h=1) |
| 3 | 5(h=4) |
| 4 | 16 |
| 5 | 44 |
| 6 | 88(h=5) |
| 7 | 12 |
| 8 | 23(h=5) |
| 9 | 13 |
| 10 | 94(h=6) |

Câu 3: Draw the 17-entry hash that results from using the hash function h(i) = (i+3) mod 17 to hash keys 1, 3, 18, 8, 23, 35, 11, 36, 20, 16.

1. Xu li xung dot theo chuoi

|  |  |
| --- | --- |
| 0 |  |
| 1 |  |
| 2 | 16 |
| 3 |  |
| 4 | 1,18,35 |
| 5 | 36 |
| 6 | 3, 20 |
| 7 |  |
| 8 |  |
| 9 | 23 |
| 10 |  |
| 11 | 8 |
| 12 |  |
| 13 |  |
| 14 | 11 |
| 15 |  |
| 16 |  |

1. Xu li xung dot theo duyet tuyen tinh

A[10] = {1, 3, 18, 8, 23, 35, 11, 36, 20, 16}

|  |  |
| --- | --- |
| 0 |  |
| 1 |  |
| 2 | 16 |
| 3 |  |
| 4 | 1 |
| 5 | 18(h=4) |
| 6 | 3 |
| 7 | 35(h=4) |
| 8 | 36(h=5) |
| 9 | 23 |
| 10 | 20(h=6) |
| 11 | 8 |
| 12 |  |
| 13 |  |
| 14 | 11 |
| 15 |  |
| 16 |  |

Câu 4:

`#include "bits/stdc++.h"

using *namespace* std;

map<*int*, string> mp;

*int* countNo(*int* *n*) {

*int* res = 0;

    while (*n*) {

        res++;

*n*/=10;

    }

    return res;

}

*int* main() {

    queue<string> q;

    string s;

    while (cin >> s) {

        q.push(s);

    }

    while (!q.empty()) {

        string s = q.front();

        q.pop();

        if (s.substr(0,6) == "Insert") {

*int* Id;

            sscanf(s.c\_str(), "Insert(%d,%[^,],%[^)])", &Id);

*int* pos = 8+countNo(Id);

*//cout << Id << " " << pos << endl;*

            mp[Id] = s.substr(pos, (*int*)(s.size()-pos-1));

        } else if (s.substr(0,5) == "Infor") {

*int* Id;

            sscanf(s.c\_str(), "Infor(%d)", &Id);

            if (mp[Id].size() > 0)

                cout << mp[Id] << endl;

            else cout << "NA,NA" << endl;

        } else if (s.substr(0,6) == "Delete") {

*int* Id;

            sscanf(s.c\_str(), "Delete(%d)", &Id);

            mp[Id] = "";

        }

    }

}