PHƯƠNG PHÁP NỐI KẾT

❖ PHƯƠNG PHÁP NỐI KẾT (CHAINING)

Phương pháp này giải quyết đụng độ bằng cách tạo một danh sách các phần tử có địa chỉ trùng nhau. Bảng băm sẽ dùng danh sách liên kết đơn tại mỗi địa chỉ của nó để nối kết các phần tử trong trường hợp đụng độ.

PHƯƠNG PHÁP NỐI KẾT

❖PHƯƠNG PHÁP NỐI KẾT (CHAINING)

Phương pháp nối kết có 2 dạng:

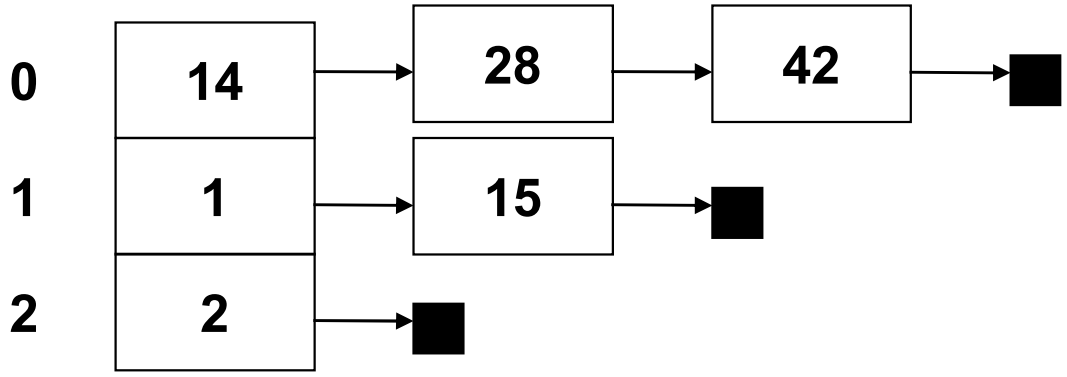
+ Nối kết trực tiếp (direct chaining): khi gặp đụng độ sẽ tạo một phần tử mới và nối vào danh sách tại vị trí có đụng độ

+ Nối kết hợp nhất (coalesced chaining): khi gặp đụng độ sẽ dùng phần tử trống đầu tiên tính từ cuối mảng để nối vào danh sách tại vị trí có đụng độ.

PHƯƠNG PHÁP NỐI KẾT

❖PHƯƠNG PHÁP NỐI KẾT (CHAINING)

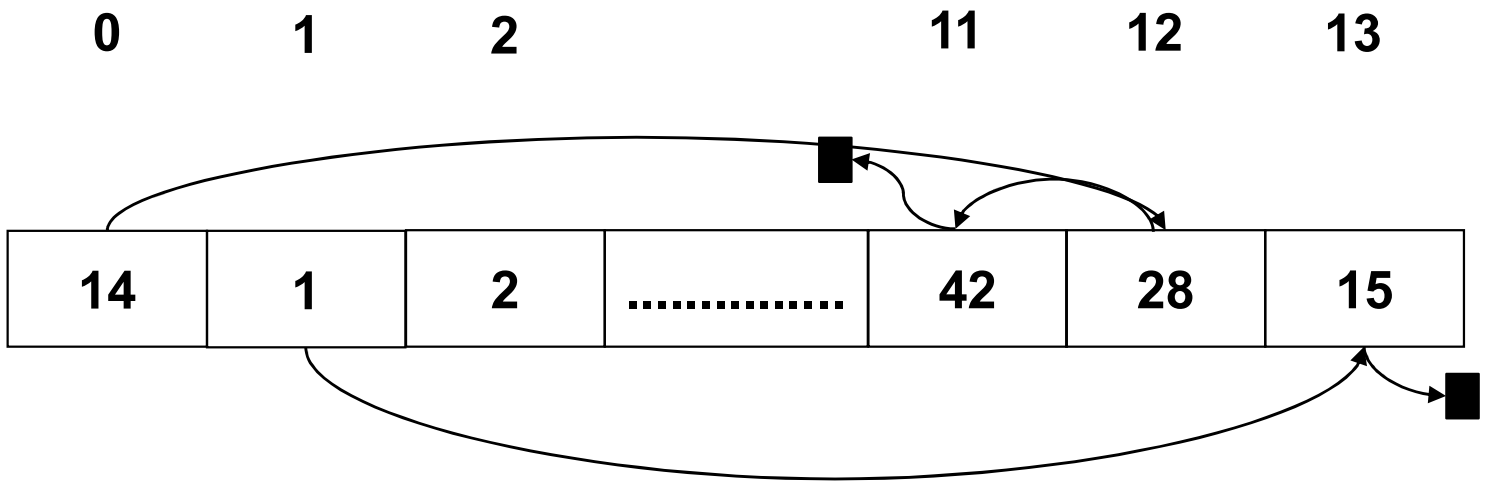
Ví dụ 7: Cho bảng băm dùng phương pháp nối kết trực tiếp với số phần tử là 14, sử dụng hàm băm theo phương pháp chia. Các khóa cần lưu trữ là 1, 2, 14, 15, 28, 42.



PHƯƠNG PHÁP NỐI KẾT

❖PHƯƠNG PHÁP NỐI KẾT (CHAINING)

Ví dụ 8: Cho bảng băm dùng phương pháp nối kết hợp nhất với số phần tử là 14, sử dụng hàm băm theo phương pháp chia. Các khóa cần lưu trữ là 1, 2, 14, 15, 28, 42.



PHƯƠNG PHÁP NỐI KẾT

❖CẤU TRÚC DỮ LIỆU

```
struct Node {
    int key; // giả sử chỉ có thông tin khóa k có kiểu int
    Node * pNext;
};
struct LIST {      Node * pHead, * pTail;   };
struct OHashtable {
    int m;
    LIST *buckets;
};
```

PHƯƠNG PHÁP NỐI KẾT

❖CÁC THAO TÁC

- Tạo bảng băm kích thước m (nối kết trực tiếp)

```
void CreateList(LIST &l);
void CreateOHashtable(OHashtable &ht, int m) {
    ht.buckets = new LIST[m];
    if (ht.buckets == NULL) ht.m = 0;
    else {
    for (int i = 0; i < m; i++) CreateList(ht.buckets[i]);
    ht.m = m;
    }
}
```

PHƯƠNG PHÁP NỐI KẾT

❖CÁC THAO TÁC

- Đưa một phần tử có khóa x (nối kết trực tiếp)

```
Node * CreateNode(int key); // tạo node có giá trị k
void AddLast(LIST &l, Node *p); // thêm cuối danh sách
int OHashCode(int key, int m); // cài đặt dạng chia hoặc nhân
int OPutKey(OHashtable ht, int key) {
    if (ht.m == 0) return 0;
    Node * p = CreateNode(key);
    if (p == NULL) return 0;
    int i = OHashCode(key, ht.m); AddLast(ht.buckets[i], p);
    return 1; }
```

PHƯƠNG PHÁP NỐI KẾT

❖CÁC THAO TÁC

- Lấy phần tử có khóa x (nối kết trực tiếp)

```
Node * Search(LIST l, int x); // hàm tìm node có khóa x
Node * OGetKey(OHashtable ht, int key) {
    if (ht.m == 0) return NULL;
    int i = OHashCode(key, ht.m);
    return Search(ht.buckets[i], key);
}
```

PHƯƠNG PHÁP NỐI KẾT

❖CÁC THAO TÁC

- Xóa phần tử có khóa x (nối kết trực tiếp)

```
int Remove(LIST &l, int x); // hàm xóa node có khóa x
int ORemoveKey(OHashtable ht, int key) {
    if (ht.m == 0) return 0;
    int i = OHashCode(key, ht.m);
    return Remove(ht.buckets[i], key);
}
```

PHƯƠNG PHÁP NỐI KẾT

❖CÁC THAO TÁC

- Hủy toàn bộ bảng băm (nối kết trực tiếp)

```
void RemoveList(LIST &l); // hàm hủy toàn bộ danh sách
void RemoveOHashtable(OHashtable &ht) {
    for (int i = 0; i < ht.m; i++) {
    RemoveList(ht.buckets[i]);
    ht.buckets[i] = NULL;
    }
    delete[] ht.buckets;
    ht.m = 0;
}
```

PHƯƠNG PHÁP NỐI KẾT

❖ĐẶC ĐIỂM

- Số phần tử không cố định
- Một số khóa có thể có cùng địa chỉ
- Có thể thực hiện thao tác thêm, xóa phần tử
- Thời gian truy xuất bị suy giảm.

PHƯƠNG PHÁP NỐI KẾT

❖BÀI TẬP

Cho mảng dữ liệu gồm các khóa có giá trị nguyên, xây dựng bảng băm theo phương pháp nối kết trực tiếp có kích thước m với hàm băm theo phương pháp chia. Viết chương trình tạo hàm băm có thể lưu 8 phần tử và nhập một dãy khóa nguyên vào bảng băm đó. In toàn bộ bảng băm và tìm và in ra phần tử có khóa x được nhập từ bàn phím.