

Cấu trúc dữ liệu - CT177

Chương 5

Tự diễn

Bộ môn Công Nghệ Phần Mềm



CANTHO UNIVERSITY

NỘI DUNG

- Khái niệm
- Cài đặt tự diễn bằng mảng
- Cài đặt tự diễn bằng bảng băm



TỰ ĐIỂN

- **Tự điển** là một kiểu dữ liệu trừu tượng **tập hợp đặc biệt** với các phép toán thêm (INSERT), bớt (DELETE) và tìm kiếm (MEMBER) có phần hiệu quả nhất.
- Tự điển có thể được cài đặt bằng:
 - Danh sách đặc (mảng).
 - Bảng băm.

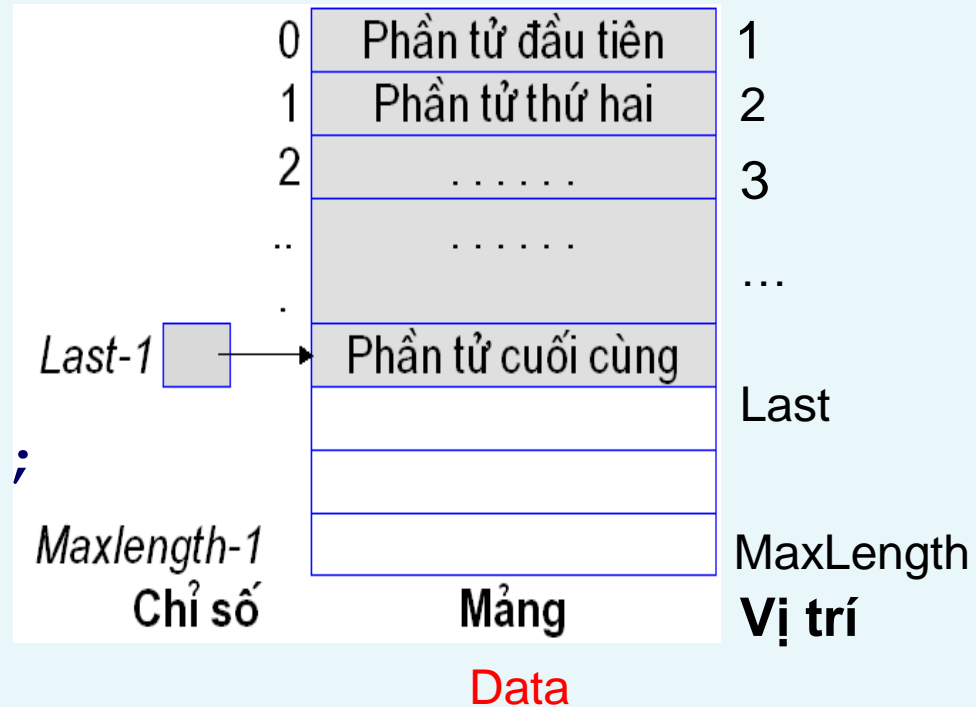


CANTHO UNIVERSITY

CÀI ĐẶT TỰ ĐIỂN BẰNG MẢNG – Danh sách đặc

- Khai báo

```
#define MaxLength ...  
typedef ... ElementType;  
typedef int Position;  
typedef struct  
{  
    ElementType Data[MaxLength];  
    Position Last;  
} SET;
```

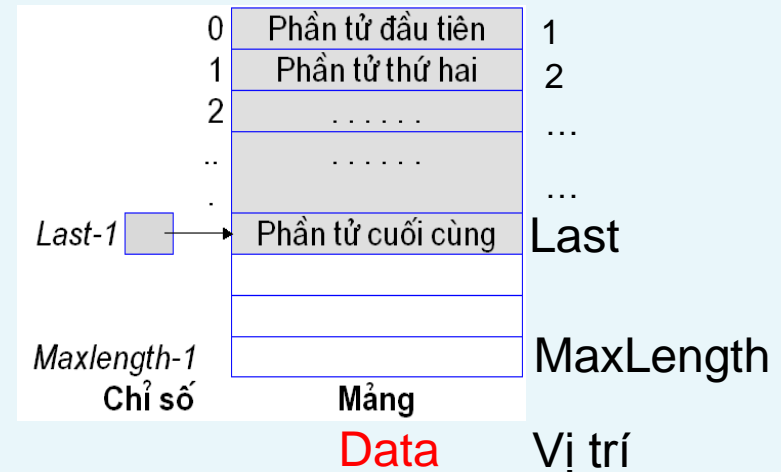




CÀI ĐẶT TỰ ĐIỂN BẰNG MẢNG

- Khởi tạo rỗng

```
void makenullSet (SET *L){  
    (*L).Last=0;  
    L->Last=0  
}
```



- Kiểm tra 1 phần tử có trong tự điển không

```
int isMember(ElementType X, SET L){  
    Position P=1, Found=0;           !Found  
    while ((P <= L.Last) && (Found == 0))  
        if ((L.Data[P-1]) == X) Found = 1;  
        else P++;  
    return Found;  
}
```



CÀI ĐẶT TỰ ĐIỂN BẰNG MẢNG

- Thêm 1 phần tử vào tự điển

```
void insertSet(ElementType X, SET *L) {  
    if (fullSet(*L)) L->Last==MaxLength  
        printf("Tap hop day"); !Member(X,*L)  
    else if (isMember(X,*L)==0) {  
        L->Last++(*L).Last++;  
        (*L).Data[( *L).Last-1]=X;  
        L->Data[L->Last-1]  
    }  
    else  
        printf("\nPt da ton tai trong t.dien");  
}
```



CÀI ĐẶT TỰ ĐIỂN BẰNG MẢNG

- Xóa 1 phần tử khỏi tự điển

```
void deleteSet (ElementType X, SET *L) {  
    if (emptySet (*L)) L->Last==0  
        printf("Tap hop rong!");  
    else{  
        Position Q=1; L->Last      L->Data  
        while ( (Q<=(*L).Last) && ( (*L).Data[Q-1] !=X) )  
            Q++;  
        if ( (*L).Data[Q-1]==X) {  
            (*L).Data[Q-1]=(*L).Data[ (*L).Last-1];  
            (*L).Last--;  
        } //if  
    }
```



CÀI ĐẶT TỰ ĐIỂN BẰNG BẰM BẰM

- **Băm (hashing)** là một kỹ thuật rất quan trọng và được dùng rộng rãi để cài đặt tự điển.
- Trong tự điển có n phần tử, cài đặt tự điển bằng **bảng băm** đòi hỏi trung bình chỉ **một hằng thời gian** cho mỗi phép toán thêm và tìm kiếm trong khi cài đặt tự điển bằng **mảng** đòi hỏi tốn **n bước** cho mỗi phép toán trên.
- Các dạng bảng băm:
 - Băm đóng
 - Băm mở



CÀI ĐẶT TỰ ĐIỂN BẰNG BẢNG BĂM

- **Hàm băm** là một ánh xạ từ tập dữ liệu A đến các số nguyên $0..B-1$. Hàm băm được sử dụng để tìm giá trị băm.
- Các phương pháp xác định hàm băm

– Phương pháp chia

x	$H(x) = x \bmod B$ với $B=10$
34	4
19	9

– Phương pháp nhân

x	x^2	H(x) gồm 3 số ở giữa
5402	29181604	181 hoặc 816
0367	00134689	134 hoặc 346

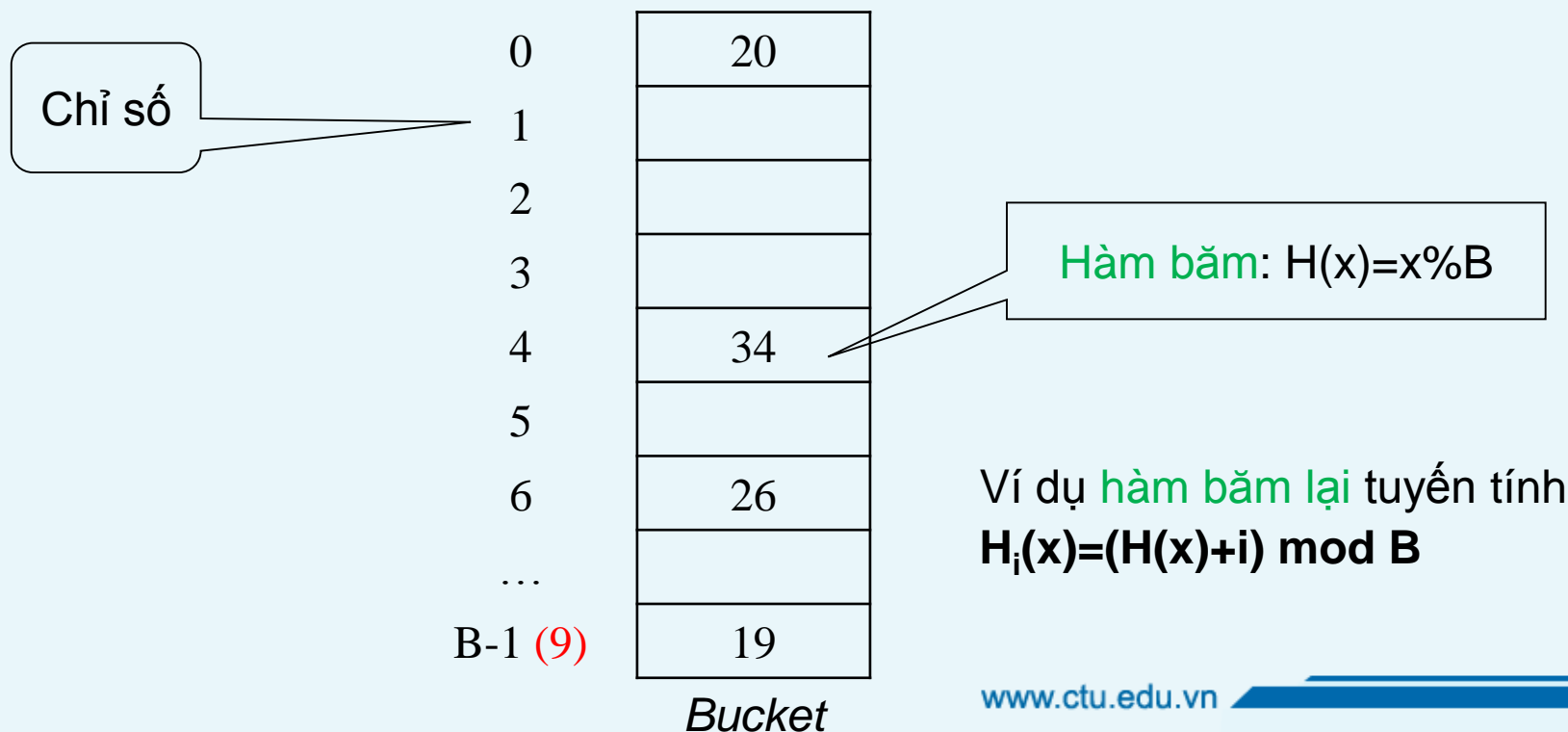
– Phương pháp tách

- Tách khóa. VD **17046329** có giá trị băm $(329+046+017)\%1000 = 392$
- Gấp khóa. VD **17046329** có giá trị băm $(923+046+710)\%1000 = 679$



BĂM ĐÓNG

- **Bảng băm đóng** lưu giữ các phần tử của tự điển ngay trong mảng.
- Bucket thứ i chứa phần tử có giá trị băm là i .
- Nếu có nhiều phần tử có cùng giá trị băm, **chiến lược băm lại** (rehash strategy) được sử dụng để giải quyết sự đụng độ.





BĂM ĐÓNG

- Khai báo

```
#define B 100
#define Deleted -1000
//Gia dinh gia tri cho o da bi xoa
#define Empty 1000
//Gia dinh gia tri cho o chua su dung
typedef int ElementType;
typedef ElementType Dictionary[B];
```



BẮM ĐÓNG

- Tạo tự điển rỗng

```
void makenullDic (Dictionary D)
```

```
{  
    for (int i=0 ; i<B; i++)  
        D[i]=Empty;  
}
```

- Hàm băm

```
int H(ElementType X)
```

```
{  
    return X%B;  
}
```

0

E

1

E

2

E

3

E

4

E

5

E

6

E

...

E

E

E

B-1 (9)



BĂM ĐÓNG

- Ví dụ: Kiểm tra xem giá trị **29**, 39 có trong bảng băm với $B=10$

Chỉ số	0	20
	1	29
	2	E
	3	E
	4	34
	5	E
	6	26
	7	E
	8	E
B-1 (9)		19

Ví dụ hàm băm lại tuyến tính

$$H_i(x) = (H(x) + i) \bmod B$$

$$i=0 \rightarrow H(29) = 9$$

$$i=1 \rightarrow H(29) = 0$$

$$i=2 \rightarrow H(29) = 1$$

$$H(x) = x \% B$$



BĂM ĐÓNG

- Kiểm tra sự tồn tại của phần tử trong tự điển

```
int isMember(ElementType X, Dictionary D)
{
    int i=0, init=H(X);
    while ((i<B) && (D[(i+init)%B] !=Empty)
        && (D[(i+init)%B] !=X))
        i++;
    return (D[(i+init)%B] ==X);
}
```



BĂM ĐÓNG

- Ví dụ: Ta cần lưu trữ các số nguyên 34, 20, 26 và 19 vào trong bảng băm có số bucket $B = 10$ và sử dụng hàm băm $h(x) = x \% 10$

Chỉ số	0	20
	1	E
	2	E
	3	E
	4	34
	5	E
	6	26
	7	E
	8	E
	B-1 (9)	19

$H(x) = x \% B$



BĂM ĐÓNG

- Ví dụ: Thêm giá trị 29 vào bảng băm có $B=10$ và sử dụng hàm băm lại $H_i(x)=(H(x)+i) \bmod B$ để giải quyết trường hợp đụng độ.

Chỉ số	0	20
	1	29
	2	E
	3	E
	4	34
	5	E
	6	26
	...	E
		E
	B-1	19

$$H(x)=x\%B$$

Ví dụ hàm băm lại tuyến tính

$$H_i(x)=(H(x)+i) \bmod B$$

$$i=0 \rightarrow H(29) = 9$$

$$i=1 \rightarrow H(29) = 0$$

$$i=2 \rightarrow H(29)=1$$



BĂM ĐÓNG

- Ví dụ: Thêm vào giá trị 30

0	20
1	29
2	12
3	30
4	34
5	E
6	26
...	E
	E
B-1	19

$i=0 \rightarrow H(30) = 0$

$i=1 \rightarrow H(30) = 1$

$i=2 \rightarrow H(30)=2$

$i=3 \rightarrow H(30)=3$



BĂM ĐÓNG

- Thêm phần tử vào tự điển

```
void insertDic(ElementType X, Dictionary D)
{
    int i=0,init;
    if (fullDic(D))
        printf("Bang bam day");
    else if (isMember(X,D)==0) {
        init=H(X);
        while ((i<B) && (D[(i+init)%B] != Empty) &&
            (D[(i+init)%B] != Deleted))
            i++;
        D[(i+init)%B]=X;
    }
    else
        printf("\nPhan tu da ton tai");
}
```



BĂM ĐÓNG

- **Bài tập:** Giả sử bảng băm có 7 bucket, hàm băm là $h(x) = x \bmod 7$. Hãy vẽ hình biểu diễn bảng băm khi ta lần lượt đưa vào bảng băm rỗng các khoá 1, 8, 27, 64, 125, 216, 343 trong trường hợp dùng bảng băm đóng với chiến lược giải quyết đụng độ là phép thử tuyến tính?



BĂM ĐÓNG

- Ví dụ: Xóa giá trị 30

0	20
1	29
2	12
3	Deleted
4	34
5	E
6	26
...	E
	E
B-1	19

$i=0 \rightarrow H(30) = 0$

$i=1 \rightarrow H(30) = 1$

$i=2 \rightarrow H(30) = 2$

$i=3 \rightarrow H(30) = 3$



BĂM ĐÓNG

- Xóa phần tử ra khỏi tự điển

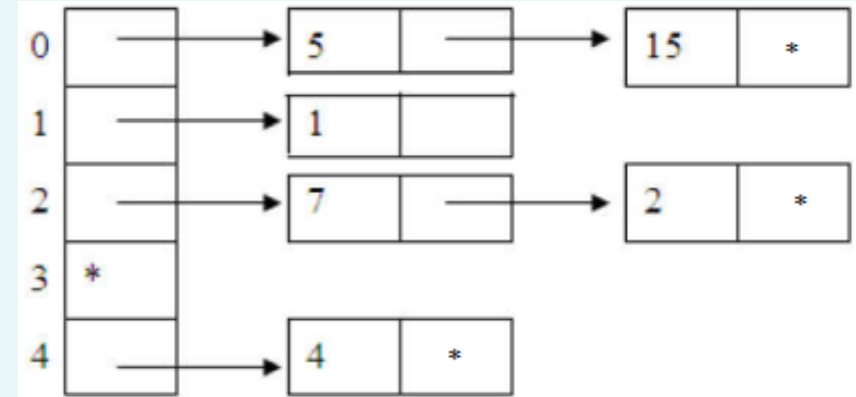
```
void deleteDic(ElementType X, Dictionary D)
{
    if (emptyDic(D))
        printf("\nBang bam rong!");
    else
    {
        int i=0,init=H(X);
        while ((i<B) && (D[(i+init)%B]!=X) &&
            (D[(i+init)%B]!=Empty))
            i++;
        if (D[(i+init)%B]==X)
            D[(i+init)%B]=Deleted;
    }
}
```



BẮM MỞ

- Khai báo

```
#define B ...  
typedef ... ElementType;  
struct Node  
{  
    ElementType Data;  
    struct Node* Next;  
};  
typedef struct Node* Position;  
typedef Position Dictionary[B];
```





BẮM MỞ

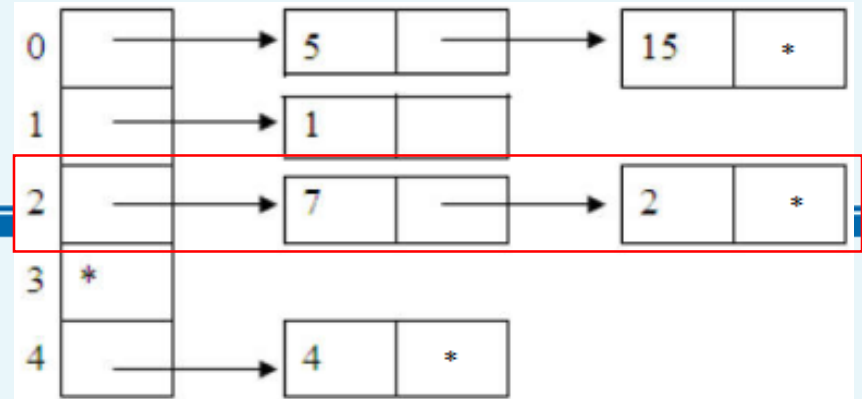
- Tạo tự điển rỗng

```
void makenullDic (Dictionary *D)
{
    for (int i=0; i<B; i++)
        (*D) [i] = NULL;
}
```

0	*
1	*
2	*
...	*
B-1	*



BĂM MỞ



- Kiểm tra sự tồn tại của một phần tử trong tự điển

```
int isMember(ElementType X, Dictionary D)
{
    Position P;
    int Found=0;
    P=D[H(X)]; //Tim o muc H(X)
    //Duyet tren ds thu H(X)
    while((P!=NULL) && (!Found))
        if (P->Data==X) Found=1;
        else P=P->Next;
    return Found;
}
```




BẮM MỞ

- Thêm phần tử vào tự điển

```
void insertDic (ElementType X, Dictionary *D)
{
    int Bucket;
    Position P;
    if (!isMember (X, *D) ) {
        Bucket=H (X) ;
        P=(*D) [Bucket] ;
        //Cap phat o nho moi cho *D[Bucket]
        (*D) [Bucket]=(struct Node*)malloc(sizeof(struct Node)) ;
        (*D) [Bucket]->Data=X;
        (*D) [Bucket]->Next=P;
    }
}
```



CANTHO UNIVERSITY

BẮM MỞ

- **Bài tập:** Giả sử bảng băm có 7 bucket, hàm băm là $h(x) = x \bmod 7$. Hãy vẽ hình biểu diễn bảng băm khi ta lần lượt đưa vào bảng băm rỗng các khoá 1, 8, 27, 64, 125, 216, 343 trong trường hợp dùng bảng băm mở?



CANTHO UNIVERSITY

BĂM
MỞ

- **Xoá phần tử ra khỏi tự điển**

```
void deleteDic(ElementType X,Dictionary *D){
    int Bucket, Done;
    Position P,Q;
    Bucket=H(X);
    if ((*D)[Bucket]!=NULL)//danh sach ton tai
    {   if ((*D)[Bucket]->Data==X) //X dau dsach
        {   Q=(*D)[Bucket];
            (*D)[Bucket]=(*D)[Bucket]->Next;
            free(Q);
        }
    }
    else // Tim X
    {   Done=0;
        P=(*D)[Bucket];
        while ((P->Next!=NULL) && (!Done))
            if (P->Next->Data==X) Done=1; else P=P->Next;
        if (Done) // Neu tim thay
        {   Q=P->Next; //Xoa P->Next
            P->Next=Q->Next;
            free(Q);
        }
    }
}
```



HẾT