


CANTHO UNIVERSITY

CHƯƠNG 2 CÁC KIỂU DỮ LIỆU TRỪU TƯỢNG CƠ BẢN (BASIC ABSTRACT DATA TYPES)

www.ctu.edu.vn 1



CANTHO UNIVERSITY

Mục tiêu

- Nắm vững các kiểu dữ liệu trừu tượng như: danh sách, ngăn xếp, hàng đợi.
- Cài đặt các kiểu dữ liệu bằng ngôn ngữ lập trình cụ thể.
- Ứng dụng được các kiểu dữ liệu trừu tượng trong bài toán thực tế.

www.ctu.edu.vn 2



NỘI DUNG SẼ HỌC

- Kiểu dữ liệu trừu tượng danh sách (LIST)
- Kiểu dữ liệu trừu tượng ngăn xếp (STACK)
- Kiểu dữ liệu trừu tượng hàng đợi (QUEUE)
- Danh sách liên kết kép (Doubly-Linked Lists)



NỘI DUNG SẼ HỌC

- Kiểu dữ liệu trừu tượng danh sách (LIST)
- Kiểu dữ liệu trừu tượng ngăn xếp (STACK)
- Kiểu dữ liệu trừu tượng hàng đợi (QUEUE)
- Danh sách liên kết kép (Doubly-Linked Lists)




CANTHO UNIVERSITY

DANH SÁCH

- Khái niệm danh sách
- Các phép toán trên danh sách
- Cài đặt danh sách
 - Dùng mảng (DS ĐẶC)
 - Dùng con trỏ (DS LIÊN KẾT)

www.ctu.edu.vn 5



CANTHO UNIVERSITY

DANH SÁCH

- Khái niệm danh sách
- Các phép toán trên danh sách
- Cài đặt danh sách
 - Dùng mảng (DS ĐẶC)
 - Dùng con trỏ (DS LIÊN KẾT)

www.ctu.edu.vn 6



CANTHO UNIVERSITY

KHÁI NIỆM VỀ DANH SÁCH

- Là tập hợp hữu hạn các phần tử có cùng kiểu
- Kiểu chung được gọi là kiểu phần tử (element type)
- Ta thường biểu diễn dạng: $a_1, a_2, a_3, \dots, a_n$
- Nếu
 - $n=0$: danh sách rỗng
 - $n>0$: phần tử đầu tiên là a_1 , phần tử cuối cùng là a_n
- Độ dài của danh sách: số phần tử của danh sách
- Các phần tử trong danh sách có thứ tự tuyến tính theo vị trí xuất hiện. Ta nói a_i đứng trước a_{i+1} ($i=1..n-1$)

www.ctu.edu.vn

7



CANTHO UNIVERSITY

DANH SÁCH

- Khái niệm danh sách
- Các phép toán trên danh sách
- Cài đặt danh sách
 - Dùng mảng (DS ĐẶC)
 - Dùng con trỏ (DS LIÊN KẾT)

www.ctu.edu.vn

8



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (1)

`insertList(x,p,L):`

- Xen phần tử x (kiểu `ElementType`) tại vị trí p (kiểu `Position`) trong danh sách L .
- Nếu vị trí p không tồn tại trong danh sách thì phép toán không được xác định.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	d	a	t	a		s	t	u	c	t	r	u	r	e	

`insertList('r',8,L)`
↓

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
	d	a	t	a		s	t	r	u	c	t	r	u	r	e	

`insertList('r',17,L)` -> phép toán không được xác định (báo lỗi và không xen 'r' vào danh sách)

www.ctu.edu.vn

9



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (2)

`locate(x,L):`

- Trả về kết quả là vị trí (kiểu `Position`) của phần tử có nội dung X trong danh sách L .
- Nếu x không có trong danh sách thì trả về `endList(L)`

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
	d	a	t	a		s	t	r	u	c	t	r	u	r	e	

- `locate('a',L)` -> 2
- `locate('s',L)` -> 6
- `locate('h',L)` -> `endList(L)`

www.ctu.edu.vn

10



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (3)

retrieve(p,L) :

- Trả về giá trị của phần tử ở vị trí p (kiểu Position) của danh sách L.
- Nếu vị trí p không có trong danh sách thì kết quả không xác định (có thể thông báo lỗi) .

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	d	a	t	a		s	t	r	u	c	t	r	u	r	e

- retrieve(6,L) -> 's'
- retrieve(16,L) -> kết quả không xác định (báo lỗi)

www.ctu.edu.vn

11



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (4)

deleteList(p,L) :

- Xoá phần tử ở vị trí p (kiểu Position) của danh sách .
- Nếu vị trí p không có trong danh sách thì phép toán không được định nghĩa và danh sách L sẽ không thay đổi.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	d	a	t	a		s	t	r	u	c	t	r	u	r	e

deleteList(12,L)

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	d	a	t	a		s	t	r	u	c	t	u	r	e

- deleteList(15,L) -> L sẽ không thay đổi.

www.ctu.edu.vn

12



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (5)

$\text{next}(p, L)$:

- vị trí của phần tử (kiểu Position) đi sau phần tử p .
- Nếu p là phần tử cuối cùng trong danh sách L thì $\text{next}(p, L)$ cho kết quả là $\text{endList}(L)$
- Nếu vị trí p không có trong danh sách thì kết quả không xác định.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	d	a	t	a		s	t	r	u	c	t	u	r	e	

- $\text{next}(6, L) \rightarrow 7$
- $\text{next}(14, L) \rightarrow \text{endList}(L)$
- $\text{next}(15, L) \rightarrow$ kết quả không xác định

www.ctu.edu.vn

13



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (6)

$\text{previous}(p, L)$:

- Vị trí của phần tử đứng trước phần tử p .
- Nếu p là phần tử đầu tiên trong danh sách thì $\text{previous}(p, L)$ không xác định.
- Nếu vị trí p không có trong danh sách thì kết quả cũng không xác định.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	d	a	t	a		s	t	r	u	c	t	u	r	e	

- $\text{previous}(6, L) \rightarrow 5$
- $\text{previous}(1, L) \rightarrow$ kết quả không xác định
- $\text{previous}(17, L) \rightarrow$ kết quả không xác định

www.ctu.edu.vn

14



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (7)

first(L) :

- Vị trí của phần tử đầu tiên trong danh sách.
- Nếu danh sách rỗng thì endList(L) được trả về.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	d	a	t	a		s	t	r	u	c	t	u	r	e	

first(L) -> 1

L	

first(L) -> endList(L)

www.ctu.edu.vn

15



CANTHO UNIVERSITY

CÁC PHÉP TOÁN (8)

emptyList (L) :

- Trả về TRUE nếu danh sách có rỗng, ngược lại nó cho giá trị FALSE.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	d	a	t	a		s	t	r	u	c	t	u	r	e	


emptyList(L) -> FALSE

L	

emptyList(L) -> TRUE

www.ctu.edu.vn


16



CÁC PHÉP TOÁN (9)


makenullList(L) :

- Khởi tạo một danh sách L rỗng .



www.ctu.edu.vn

17




CÁC PHÉP TOÁN (10)

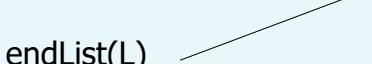
endList(L) :

- Trả về vị trí sau phần tử cuối cùng trong danh sách L.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	d	a	t	a		s	t	r	u	c	t	u	r	e	




first (L) -> endList(L)



www.ctu.edu.vn

18



CÁC PHÉP TOÁN (11)


printList(L) :

- Hiển thị giá trị các phần tử trong danh sách L theo thứ tự.

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	d	a	t	a		s	t	r	u	c	t	u	r	e	

printList(L) -> data structure

www.ctu.edu.vn
19



VÍ DỤ

Dùng các phép toán trừu tượng trên danh sách, viết chương trình con nhận vào 1 danh sách rồi sắp xếp danh sách theo thứ tự tăng dần

```

void sort(List L){
    Position p,q;    //kiểu vị trí của các phần tử trong danh sách
    p= first(L); //vị trí phần tử đầu tiên trong danh sách
    while (p!=endList(L)){
        q=next(p,L); //vị trí phần tử đứng ngay sau phần tử p
        while (q!=endList(L)){
            if (retrieve(p,L) > retrieve(q,L))
                swap(p,q); // hoán đổi nội dung 2 phần tử
            q=next(q,L);
        }
        p=next(p,L);
    }
}
  
```

www.ctu.edu.vn
20



CANTHO UNIVERSITY

Câu hỏi

- Muốn thêm phần tử x vào đầu hay cuối danh sách ta gọi phép toán nào và gọi phép toán đó như thế nào?

www.ctu.edu.vn

21



CANTHO UNIVERSITY

Trả lời

- Thêm phần tử x vào đầu danh sách

`insertList(x, first(L), L)`

- Thêm phần tử x vào cuối danh sách

`insertList(x, endList(L), L)`

www.ctu.edu.vn

22



Bài tập

- Sử dụng các phép toán trừu tượng trên danh sách, hãy viết hàm `delete_duplicate(LIST L)` loại bỏ những giá trị trùng lặp trong danh sách.

– VD: L: data structure

`delete_duplicate(L)`

→ L: data structure

23

www.ctu.edu.vn




Trả lời

```
void delete_duplicate(List L)
{
    Position p,q;    //kiểu vị trí của các phần tử trong danh sách
    p=first(L); //vị trí phần tử đầu tiên trong danh sách
    while (p!=endList(L))
    {
        q=next(p,L);    //vị trí phần tử đứng ngay sau phần tử p
        while (q!=endList(L))
        {
            if (retrieve(p,L) == retrieve(q,L))
                deleteList(q,L); //xóa phần tử
            else
                q=next(q,L);
        }
        p=next(p,L);
    }
}
```

24

www.ctu.edu.vn




CANTHO UNIVERSITY

CHƯƠNG 2 CÁC KIỂU DỮ LIỆU TRỪU TƯỢNG CƠ BẢN (BASIC ABSTRACT DATA TYPES)

Phần 2

25

www.ctu.edu.vn




CANTHO UNIVERSITY

DANH SÁCH

- Khái niệm danh sách
- Các phép toán trên danh sách
- Cài đặt danh sách
 - Dùng mảng (DS ĐẶC)
 - Dùng con trỏ (DS LIÊN KẾT)

26


www.ctu.edu.vn



CÀI ĐẶT DANH SÁCH BẰNG MẢNG (DANH SÁCH ĐẶC)

- Dùng 1 mảng để lưu trữ liên tiếp các phần tử, bắt đầu từ vị trí đầu tiên
- Ta phải ước lượng số phần tử tối đa của danh sách
- Ta phải lưu trữ độ dài hiện tại của danh sách (Last)

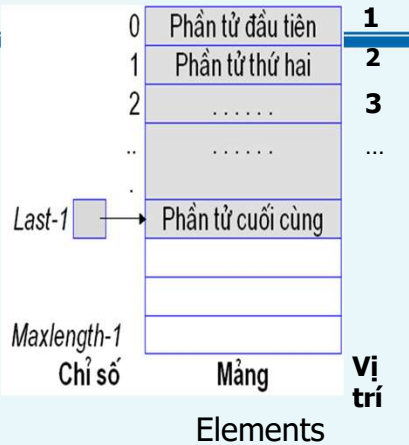
www.ctu.edu.vn
27



KHAI BÁO


```

//Độ dài tối đa của danh sách
#define MaxLength 300
//kiểu của phần tử trong danh sách
typedef int ElementType;
typedef int Position;
//kiểu vị trí của các phần tử
typedef struct {
    //mảng chứa các phần tử của danh sách
    ElementType Elements[MaxLength];
    //giữ độ dài danh sách
    Position Last;
} List;
List L;
        
```



Ta định nghĩa vị trí của một phần tử trong danh sách là “chỉ số của mảng tại vị trí lưu trữ phần tử đó + 1”

www.ctu.edu.vn
28



KHỞI TẠO DANH SÁCH RỖNG

Last= 0 0
 1
 Maxlength-1


Chỉ số
Mảng

- Cho độ dài danh sách bằng 0

```

void makenullList(List *pL) {
    pL->Last=0;
}
    
```

29
www.ctu.edu.vn



KIỂM TRA DANH SÁCH RỖNG

- Xem độ dài danh sách có bằng 0 không?

```

int emptyList(List L) {
    return L.Last==0;
}
    
```

30
www.ctu.edu.vn



CANTHO UNIVERSITY

KIỂM TRA DANH SÁCH ĐẦY

- Xem độ dài danh sách có bằng MaxLength không?

```
int fullList(List L){
    return L.Last==MaxLength;
}
```

www.ctu.edu.vn

31



CANTHO UNIVERSITY

XEN PHẦN TỬ X VÀO VỊ TRÍ P (1)

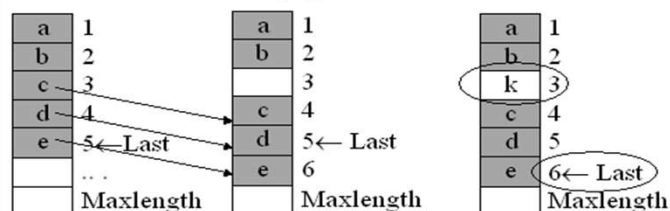
- Xen phần tử $x='k'$ vào vị trí $p=3$ trong danh sách L (chỉ số 2 trong mảng)

- Nếu L có dạng :

⇒ Trường hợp này mảng đầy ⇒ báo lỗi

Chỉ số mảng	Nội dung	Vị trí trong dsách
0	a	1
1	b	2
2	c	3
3	d	4
4	e	5
5	h	Maxlength ← Last

- Nếu danh sách L có dạng :



- Đời các phần tử từ vị trí 3 đến cuối danh sách ra sau một vị trí
- Độ dài danh sách tăng 1
- Đưa k vào vị trí 3

32



CANTHO UNIVERSITY

XEN PHẦN TỬ X VÀO VỊ TRÍ P (2)

- Thuật toán:

Để chèn x vào vị trí p của L, ta làm như sau:

- Nếu mảng đầy thì thông báo lỗi
- Ngược lại, nếu vị trí p không hợp lệ thì báo lỗi
- Ngược lại:
 - Dời các phần tử từ vị trí p đến cuối danh sách ra sau một vị trí
 - Đưa phần tử mới x vào tại vị trí p
 - Độ dài danh sách tăng 1

33

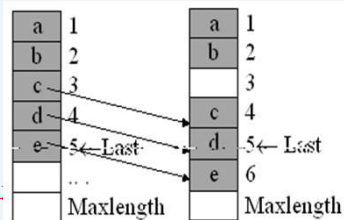
www.ctu.edu.vn



CANTHO UNIVERSITY


XEN PHẦN TỬ X VÀO VỊ TRÍ P (3)

```
void insertList(ElementType X, Position P, List *pL) {
    if (pL->Last == MaxLength)
        printf("Danh sach day");
    else if (P < first(*pL) || P > endList(*pL))
        printf("Vi tri khong hop le");
    else {
        Position Q;
        /*Dời các phần tử từ vị trí p
        đến cuối dsách ra trước 1 vị trí*/
        for (Q = pL->Last; Q > P - 1; Q--)
            pL->Elements[Q] = pL->Elements[Q - 1];
        //Tăng độ dài danh sách lên 1
        pL->Last++;
        //Đưa x vào vị trí p
        pL->Elements[P - 1] = X;
    }
}
```



34

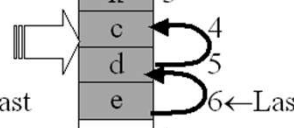
www.ctu.edu.vn



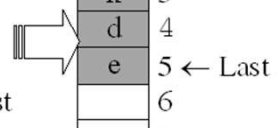
XÓA MỘT PHẦN TỬ TẠI VỊ TRÍ P TRONG DS (1)

• Ví dụ: Xóa phần tử vị trí $p=4$ của L

CS	NỘI DUNG	VỊ TRÍ
0	a	1
1	b	2
2	k	3
3	c	4
4	d	5
5	e	6 ← Last
...		
		Maxlength




a	1
b	2
k	3
c	4
d	5
e	6 ← Last
...	
	Maxlength



a	1
b	2
k	3
d	4
e	5 ← Last
	6
...	
	Maxlength

35

www.ctu.edu.vn



XÓA MỘT PHẦN TỬ TẠI VỊ TRÍ P TRONG DS (2)

- Nếu p là một vị trí không hợp lệ thì thông báo lỗi
- Ngược lại:
 - Di dời các phần tử từ vị trí $p+1$ đến cuối danh sách ra trước một vị trí
 - Độ dài của danh sách giảm 1

36

www.ctu.edu.vn



XÓA MỘT PHẦN TỬ TẠI VỊ TRÍ P TRONG DS (3)

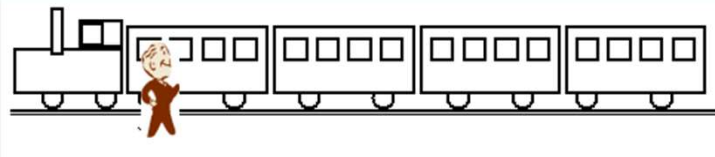
```
void deleteList(Position P, List *pL){
    if (P<1 || P>pL->Last)
        printf("Vị trí không hợp lệ");
    else if (emptyList(*pL))
        printf("Danh sách rỗng!");
    else{
        Position Q;
        /*Dời các phần tử từ vị trí p+1 đến cuối
        danh sách ra trước 1 vị trí*/
        for (Q=P-1; Q<pL->Last-1; Q++)
            pL->Elements[Q]=pL->Elements[Q+1];
        pL->Last--;
    }
}
```

www.ctu.edu.vn

37



TÌM KIẾM PHẦN TỬ X TRONG DS(1)



- Bắt đầu từ phần tử đầu tiên trong danh sách, ta tiến hành tìm từ đầu danh sách cho đến khi tìm thấy hoặc cuối danh sách
 - Nếu giá trị tại vị trí P bằng X
 $\text{retrieve}(P, L) == X$
 thì dừng tìm kiếm
 - Ngược lại (giá trị tại vị trí P khác X) thì đến vị trí kế tiếp
 $P = \text{next}(P, L)$
- Trả về vị trí phần tử được tìm thấy hoặc vị trí Last+1(endList) nếu không tìm thấy.

www.ctu.edu.vn

38



CANTHO UNIVERSITY

TÌM KIẾM PHẦN TỬ X TRONG DS(1)

```

Position locate(ElementType X, List L){
    Position P;
    int Found = 0;
    P = first(L); //vị trí phần tử đầu tiên
    /*trong khi chưa tìm thấy và chưa kết
    thúc danh sách thì xét phần tử kế tiếp*/
    while ((P != endList(L)) && (Found == 0))
        if (retrieve(P,L) == X) Found = 1;
        else P = next(P, L);
    return P;
}

```

www.ctu.edu.vn

39



CANTHO UNIVERSITY

ĐÁNH GIÁ GIẢI THUẬT TÌM KIẾM

- Thời gian tìm kiếm
 - nhanh nhất (tốt nhất) là khi nào, x ở đâu?
 - xấu nhất khi nào?
- Độ phức tạp của giải thuật thường được xác định là trong trường hợp xấu nhất $O(n)$

www.ctu.edu.vn

40



CANTHO UNIVERSITY

CÁC PHÉP TOÁN KHÁC (1)

- Xác định vị trí sau phần tử cuối trong danh sách

```
Position endList(List L){
    return L.Last+1;
}
```

- Xác định vị trí đầu tiên trong danh sách

```
Position first(List L){
    return 1;
}
```

www.ctu.edu.vn

41



CANTHO UNIVERSITY

CÁC PHÉP TOÁN KHÁC (2)

- Xác định nội dung phần tử tại vị trí P trong danh sách

```
ElementType retrieve(Position P, List L){
    return L.Elements[P-1];
}
```

- Xác định vị trí kế tiếp trong danh sách

```
Position next(Position P, List L){
    return P+1;
}
```

www.ctu.edu.vn

42



CANTHO UNIVERSITY

CÁC PHÉP TOÁN KHÁC (2)

- In danh sách, giả sử danh sách số nguyên

```
void printList(List L){
    Position P= first(L);
    while (P != endList(L)){
        printf("%d ", L.Elements[P]);
        P=next(P,L);
    }
}
```

www.ctu.edu.vn

43




CANTHO UNIVERSITY

BÀI TẬP

- Vận dụng các phép toán trên danh sách để viết chương trình nhập vào một danh sách các số nguyên và hiển thị danh sách vừa nhập ra màn hình.
- Thêm phần tử có nội dung x vào danh sách tại vị trí p (trong đó x và p được nhập từ bàn phím).
- Xóa phần tử đầu tiên có nội dung x (nhập từ bàn phím) ra khỏi danh sách.
- Sử dụng các phép toán trừu tượng trên danh sách, hãy viết hàm delete_duplicate(LIST L) loại bỏ những giá trị trùng lặp trong danh sách.

www.ctu.edu.vn


44



Trả lời bài tập

CANTHO UNIVERSITY

www.ctu.edu.vn 45




DANH SÁCH

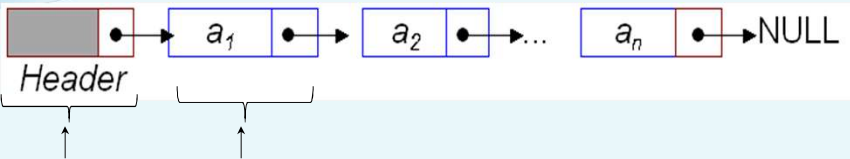
CANTHO UNIVERSITY

- Khái niệm danh sách
- Các phép toán trên danh sách
- Cài đặt danh sách
 - Dùng mảng (DS ĐẶC)
 - Dùng con trỏ (DS LIÊN KẾT)

www.ctu.edu.vn 46




CÀI ĐẶT DANH SÁCH BẢNG CON TRỎ



Header Header->Next

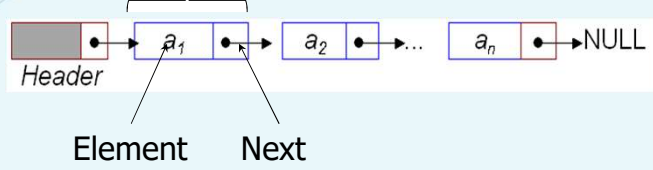
- Nối kết các phần tử liên tiếp nhau bằng con trỏ
 - Phần tử a_i trỏ tới phần tử a_{i+1}
 - Phần tử a_n trỏ tới phần tử đặc biệt NULL
 - Phần tử header trỏ tới phần tử đầu tiên a_1

www.ctu.edu.vn
47



CÀI ĐẶT DANH SÁCH BẢNG CON TRỎ

Mô hình Node



Element Next

```

typedef <DataType> ElementType; //kiểu của phần tử trong danh sách
struct Node{
    ElementType Element;    //Chứa nội dung của phần tử
    struct Node* Next;      //con trỏ chỉ đến phần tử kế tiếp
};
typedef struct Node* Position; //Kiểu vị trí
typedef Position List;
  
```

www.ctu.edu.vn
48



KHỞI TẠO DANH SÁCH RỖNG

- Cấp phát vùng nhớ cho Header
- Cho trường Next của Header trở về NULL


```

void makenullList(List *pL){
    (*pL)=(Node*)malloc(sizeof(struct Node));
    (*pL)->Next= NULL;
}

```



www.ctu.edu.vn
49



KIỂM TRA DANH SÁCH RỖNG


- Xem trường Next của ô Header có trở về NULL hay không?

```

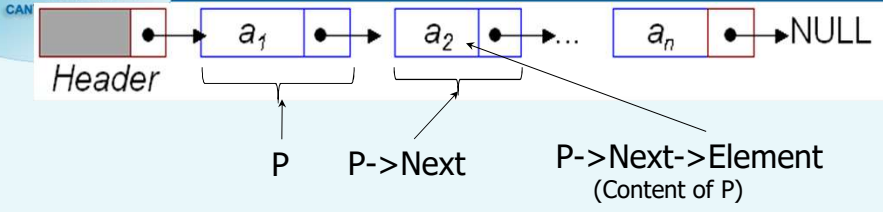
int emptyList(List L){
    return (L->Next==NULL);
}

```

www.ctu.edu.vn
50



XÁC ĐỊNH NỘI DUNG PHẦN TỬ



```


ElementType retrieve(Position P, List L){
    if (P->Next!=NULL)
        return P->Next->Element;
}

```

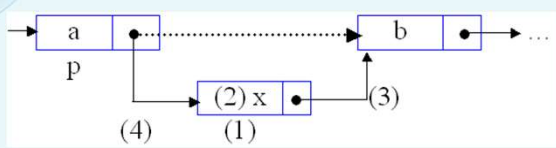
So sánh với cách cài đặt bằng mảng

www.ctu.edu.vn

51



XEN MỘT PHẦN TỬ VÀO DANH SÁCH



- Để xen phần tử x vào vị trí p của L, ta làm như sau:
 - Cấp phát 1 ô mới để lưu trữ phần tử x
 - Nối kết lại các con trỏ để đưa ô mới này vào vị trí p

```

void insertList(ElementType X, Position P, List *pL){
    Position T;
    T=(struct Node*)malloc(sizeof(struct Node));
    T->Element=X;
    T->Next=P->Next;
    P->Next=T;
}

```

Cho nhận xét đánh giá độ phức tạp so với cách dùng mảng

52



CANTHO UNIVERSITY

Câu hỏi

- Nếu khi gọi hàm

```
insertList(ElementType X, Position P, List *pL)
```

ta truyền giá trị NULL thì có ảnh hưởng gì đến kết quả trả về của hàm không? Tại sao?

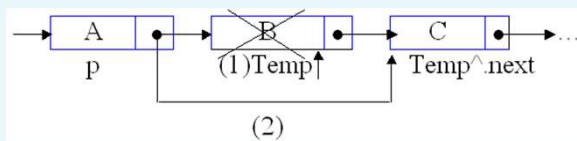
www.ctu.edu.vn

53



CANTHO UNIVERSITY

XÓA MỘT PHẦN TỬ KHỎI DANH SÁCH(1)




=> Muốn xóa phần tử ở vị trí p trong danh sách ta cần nối kết lại các con trỏ bằng cách cho p trỏ tới phần tử đứng sau phần tử thứ p

```
void deleteList(Position P, List *pL){
    Position Temp;
    if (P->Next!=NULL){
        //giữ ô chứa phần tử bị xóa để thu hồi vùng nhớ
        Temp=P->Next;
        //nối kết con trỏ trỏ tới phần tử kế tiếp
        P->Next=Temp->Next;
        //thu hồi vùng nhớ
        free(Temp);
    }
}
```

Cho nhận xét đánh giá độ phức tạp so với cách dùng mảng

54




XÁC ĐỊNH VỊ TRÍ PHẦN TỬ

- Vị trí phần tử đầu tiên
 Position first(List L){
 return L;
 }
- Vị trí sau phần tử cuối cùng
 Position endList(List L){
 Position P;
 P=First(L);
 while (P->Next!=NULL)
 P=P->Next;
 return P;
 }
- Vị trí phần tử kế tiếp
 Position Next(Position P, List L){
 return P->Next;
 }

Cho nhận xét đánh giá độ phức tạp so với cách dùng mảng


www.ctu.edu.vn
55



Câu hỏi

- Hãy thiết kế hàm locate bằng cách sử dụng các phép toán trừu tượng cơ bản trên danh sách?

www.ctu.edu.vn
56



TÌM KIẾM MỘT PHẦN TỬ TRONG DANH SÁCH


- Bắt đầu từ phần tử đầu tiên trong danh sách, ta tiến hành tìm từ đầu danh sách cho đến khi tìm thấy hoặc cuối danh sách
 - Nếu giá trị tại vị trí P bằng X
retrieve(P,L) == X hay **P->next->element == x**
 thì dừng tìm kiếm
 - Ngược lại (giá trị tại vị trí P khác X) thì đến vị trí kế tiếp
P = next(P,L)
- Trả về vị trí phần tử được tìm thấy hoặc vị trí EndList nếu không tìm thấy.

```

Position locate(ElementType X, List L){
    Position P;
    int Found = 0;
    P = first(L);
    while ((next(P,L) != NULL) && (Found == 0))
        if (retrieve(P,L) == X) Found = 1;
        else P = next(P,L);
    return P;
}
            
```

57

Cài đặt lại hàm Locate bằng cách loại bỏ biến Found.



TÌM KIẾM MỘT PHẦN TỬ TRONG DANH SÁCH


```

Position locate(ElementType X, List L){

    Position P;
    P = first(L);
    while (next(P,L) != NULL)
        if (retrieve(P,L) == X)
            return P;
        else P = next(P,L);
    return P; // endList(L)
}
            
```

58

www.ctu.edu.vn



TÌM KIẾM MỘT PHẦN TỬ TRONG DANH SÁCH

CANTHO UNIVERSITY

```

Position locate(ElementType X, List L){
    Position P;
    P = first(L);
    while (next(P,L) != NULL)
        if (retrieve(P,L) == X)
            return P;
        else P = next(P,L);
    return P; // endList(L)
}

```


?

```

Position locate(ElementType X, List L){
    Position P;
    P = first(L);
    while (P!= endList(L))
        if (retrieve(P,L) == X) return P;
        else P = next(P,L);
    return P;
}

```

www.ctu.edu.vn
59



TÌM KIẾM MỘT PHẦN TỬ TRONG DANH SÁCH

CANTHO UNIVERSITY

Cài đặt hàm

Position myLocate(ElementType x, int i, List L)

Trả về vị trí của lần xuất hiện thứ i của x trong L.

www.ctu.edu.vn
60



CANTHO UNIVERSITY

TÌM KIẾM MỘT PHẦN TỬ TRONG DANH SÁCH

```

Position myLocate(ElementType x, int i, List L){
    Position p= first(L);
    int count =0;
    while (Next(p,L) != NULL && count < i){
        if (retrieve(p,L) == x)
            count++;
        if (count<i)
            p=next(p,L);
    }
    return p;
}

```

www.ctu.edu.vn

61



CANTHO UNIVERSITY

IN DANH SÁCH RA MÀN HÌNH

```

void printList(List L){
    Position P;
    P = first(L);
    while (P != endList(L)){
        printf("%d ", retrieve(P,L));
        P = next(P, L);
    }
    printf("\n");
}

```

www.ctu.edu.vn

62



CANTHO UNIVERSITY

Câu hỏi

Hãy Cài đặt phép toán

Position Previous(Position P, List L)

bằng cách sử dụng các phép toán trừu tượng cơ bản trên danh sách?

www.ctu.edu.vn

63



CANTHO UNIVERSITY

Previous(Position P, List L)

```
Position Previous(Position P, List L){
    Position Q;
    Q= first(L);
    while (next(Q,L)!= NULL) //
        if (next(Q,L) == P)
            return Q;
        else
            Q = next(Q,L);
    return NULL;
}
```

Nếu danh sách có nhiều giá trị trùng (Ví dụ: 4 7 3 4 6 7 ...) thì
 khi tìm các vị của lần xuất hiện sau thì kết quả phép toán trên có đúng không?
 ->Viết đoạn chương trình để kiểm tra
 Có thể cải thiện tốc độ của đoạn chương trình trên?



CANTHO UNIVERSITY

SO SÁNH 2 PHƯƠNG PHÁP CÀI ĐẶT DS

- Bạn hãy phân tích ưu và khuyết điểm của
 - Danh sách đặc
 - Danh sách liên kết
- Bạn nên chọn pp cài đặt nào cho ứng dụng của mình?

www.ctu.edu.vn

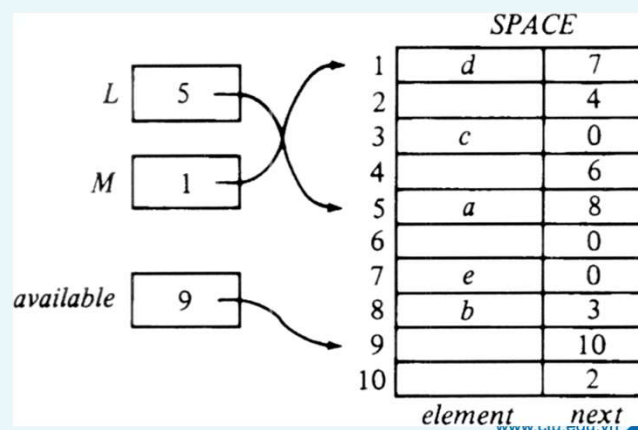
65



CANTHO UNIVERSITY

Đọc thêm

Cài đặt danh sách bằng con nháy



66



CANTHO UNIVERSITY

BÀI TẬP

Vận dụng các phép toán trên danh sách liên kết để viết chương trình:

- Nhập vào một danh sách các số nguyên
- Hiển thị danh sách vừa nhập ra màn hình
- Thêm phần tử có nội dung x vào danh sách tại vị trí p (trong đó x và p được nhập từ bàn phím)
- Xóa phần tử đầu tiên có nội dung x (nhập từ bàn phím) ra khỏi danh sách
- Viết hàm

Position myLocate(ElementType x, int i, List L)

67

www.ctu.edu.vn


CANTHO UNIVERSITY

Trả lời bài tập

```
Position myLocate(ElementType x, int i, List L){
    Position p= first(L);
    int count =0;
    while (next(p,L) != NULL && count < i){
        if (retrieve(p,L) == x)
            count++;
        if (count<i)
            p=next(p,L);
    }
    return p;
}
```

68

www.ctu.edu.vn




NỘI DUNG SẼ HỌC

- Kiểu dữ liệu trừu tượng danh sách (LIST)
- Kiểu dữ liệu trừu tượng ngăn xếp (STACK)
- Kiểu dữ liệu trừu tượng hàng đợi (QUEUE)
- Danh sách liên kết kép (Double – Lists)



NGĂN XẾP (STACK)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT
 - CÀI ĐẶT BẰNG DANH SÁCH LIÊN KẾT
 - CÀI ĐẶT BẰNG MẢNG




NGĂN XẾP (STACK)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT
 - CÀI ĐẶT BẰNG DANH SÁCH LIÊN KẾT
 - CÀI ĐẶT BẰNG MẢNG

71

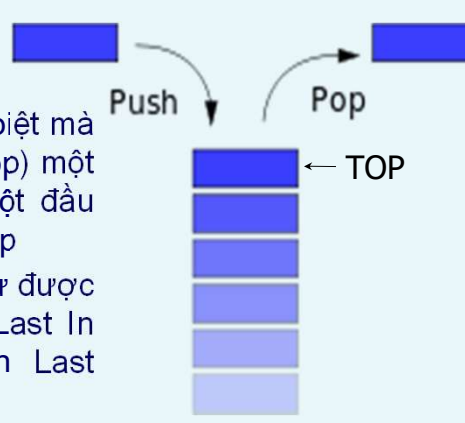
www.ctu.edu.vn



ĐỊNH NGHĨA


Ngăn xếp:

- Là một dạng danh sách đặc biệt mà việc thêm (Push) hay xóa (Pop) một phần tử chỉ thực hiện tại một đầu gọi là đỉnh (TOP) của ngăn xếp
- Việc thêm và xóa một phần tử được thực hiện theo dạng LIFO (Last In First Out) hay FILO (First In Last Out)



72

www.ctu.edu.vn




NGĂN XẾP (STACK)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT
 - CÀI ĐẶT BẰNG DANH SÁCH LIÊN KẾT
 - CÀI ĐẶT BẰNG MẢNG

73

www.ctu.edu.vn




CÁC PHÉP TOÁN

Phép toán	Diễn giải
makenullStack(S)	Tạo một ngăn xếp rỗng (S)
emptyStack(S)	Kiểm tra xem ngăn xếp S có rỗng hay không. Hàm cho kết quả 1 (true) nếu ngăn xếp rỗng và 0 (false) trong trường hợp ngược lại.
full(S)	Kiểm tra xem ngăn xếp S có đầy hay không
push(X,S)	Thêm phần tử X vào đỉnh ngăn xếp S. Tương đương: INSERT(X,FIRST(S),S)
pop(S)	Xóa phần tử tại đỉnh ngăn xếp S. Tương đương: DELETE(FIRST(S),S)
top(S)	Trả về phần tử đầu tiên trên đỉnh ngăn xếp S, tương đương: RETRIEVE(FIRST(S),S)

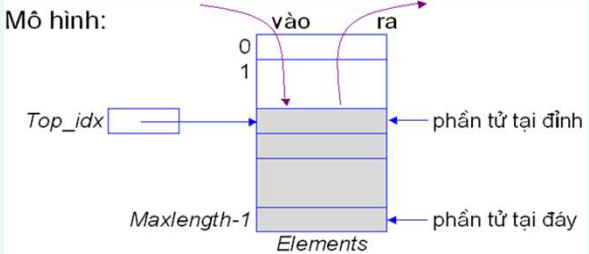
74

www.ctu.edu.vn



CÀI ĐẶT NGĂN XẾP BẰNG MẢNG (1)

Mô hình:




- Khai báo

```

#define MaxLength ...//độ dài của mảng
typedef ... ElementType;//kiểu phần tử của ngăn xếp
typedef struct {
    //Lưu nội dung của các phần tử
    ElementType Elements[MaxLength];
    int Top_idx; //giữ vị trí đỉnh ngăn xếp
}Stack;

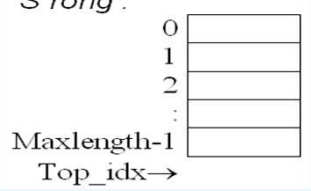
```

www.ctu.edu.vn
75



KHỞI TẠO NGĂN XẾP RỖNG

S rỗng :



– Khi ngăn xếp S rỗng ta cho đỉnh ngăn xếp được khởi tạo bằng Maxlength

```

void makenullStack(Stack *pS){
    pS->Top_idx=MaxLength;
}

```

www.ctu.edu.vn
76



CANTHO UNIVERSITY

KIỂM TRA NGĂN XẾP RỖNG?

- Ta kiểm tra xem đỉnh ngăn xếp có bằng MaxLength không?

```
int emptyStack(Stack S){
    return S.Top_idx==MaxLength;
}
```

www.ctu.edu.vn

77



CANTHO UNIVERSITY

KIỂM TRA NGĂN XẾP ĐẦY?

S đây :

Top_idx→0

1

2

:

Maxlength-1




- Ta kiểm tra xem Top_idx có chỉ vào 0 hay không?

```
int fullStack(Stack S){
    return S.Top_idx==0;
}
```

www.ctu.edu.vn

78



TRẢ VỀ PHẦN TỬ ĐẦU NGĂN XẾP

Ví dụ :

0
1
Top_idx→2
3
:
Maxlength-1

x


- Giải thuật :** Kết quả của phép toán trên ngăn xếp là x
- Nếu ngăn xếp rỗng thì thông báo lỗi
- Ngược lại, trả về giá trị được lưu trữ tại ô có chỉ số là Top_idx

```

ElementType top(Stack S) {
    if (!emptyStack(S))
        return S.Elements[S.Top_idx];
    else printf("Loi! Ngan xep rong");
}
  
```

www.ctu.edu.vn

79



XOA PHẦN TỬ TẠI ĐỈNH NGĂN XẾP

Ví dụ :

0
1
Top_idx→2
3
:
Maxlength-1

x

⇒

0
1
2
Top_idx→3
:
Maxlength-1


- Giải thuật .**
- Nếu ngăn xếp rỗng thì thông báo lỗi
- Ngược lại, tăng Top_idx lên 1 đơn vị

```

void pop(Stack *pS){
    if (!emptyStack(*pS))
        pS->Top_idx=pS->Top_idx+1;
    else printf("Loi! Ngan xep rong!");
}
  
```

www.ctu.edu.vn

80



THÊM PHẦN TỬ X VÀO NGĂN XẾP

Ví dụ :

0	
1	
Top_idx→2	
3	
⋮	
Maxlength-1	

⇒

0	
Top_idx→1	x
2	
3	
⋮	
Maxlength-1	


Giải thuật :

- Nếu ngăn xếp đầy thì thông báo lỗi
- Ngược lại, giảm Top_idx xuống 1 đơn vị rồi đưa giá trị x vào ô có chỉ số Top_idx

```

void push(ElementType X, Stack *pS){
    if (fullStack(*pS))
        printf("Loi! Ngan xep day!");
    else{
        pS->Top_idx=pS->Top_idx-1;
        pS->Elements[pS->Top_idx]=X;
    }
}
  
```


81
www.ctu.edu.vn



BÀI TẬP

- Viết hàm đổi số nguyên n sang số nhị phân (có sử dụng các phép toán trên ngăn xếp)


82
www.ctu.edu.vn



Trả lời bài tập

CANTHO UNIVERSITY


www.ctu.edu.vn 83



Trả lời bài tập

CANTHO UNIVERSITY

www.ctu.edu.vn 84




NỘI DUNG SẼ HỌC

- Kiểu dữ liệu trừu tượng danh sách (LIST)
- Kiểu dữ liệu trừu tượng ngăn xếp (STACK)
- Kiểu dữ liệu trừu tượng hàng đợi (QUEUE)
- Danh sách liên kết kép (Double – Lists)

85

www.ctu.edu.vn




HÀNG ĐỢI (QUEUE)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT HÀNG ĐỢI
 - DÙNG MẢNG DI CHUYỂN TỊNH TIẾN
 - DÙNG MẢNG VÒNG
 - DÙNG DSLK

86


www.ctu.edu.vn



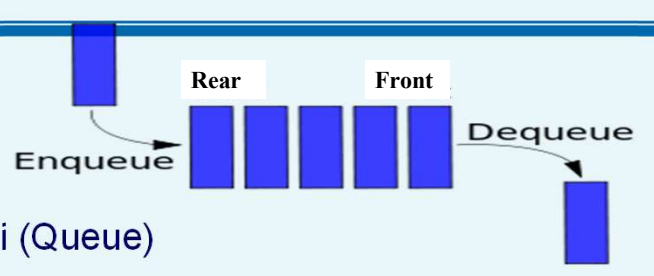
HÀNG ĐỢI (QUEUE)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT HÀNG ĐỢI
 - DÙNG MẢNG DI CHUYỂN TỊNH TIẾN
 - DÙNG MẢNG VÒNG
 - DÙNG DSLK

www.ctu.edu.vn
87



ĐỊNH NGHĨA HÀNG ĐỢI



Hàng đợi (Queue)

- Là một dạng danh sách đặc biệt, mà phép thêm vào (enQueue) được thực hiện ở đầu cuối hàng (REAR), còn phép loại bỏ (deQueue) được thực hiện ở đầu kia của danh sách, gọi là đầu hàng(FRONT)
- Cách làm việc theo dạng FIFO (First In First Out)

www.ctu.edu.vn
88



CANTHO UNIVERSITY

HÀNG ĐỢI (QUEUE)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT HÀNG ĐỢI
 - DÙNG MẢNG DI CHUYỂN TỊNH TIẾN
 - DÙNG MẢNG VÒNG
 - DÙNG DSLK

www.ctu.edu.vn

89



CANTHO UNIVERSITY

CÁC PHÉP TOÁN

Phép toán	Diễn giải
<code>makeNullQueue(Q)</code>	Tạo một hàng đợi rỗng (Q)
<code>emptyQueue(Q)</code>	Kiểm tra xem hàng đợi Q có rỗng không
<code>fullQueue(Q)</code>	Kiểm tra hàng đầy
<code>enqueue(X, Q)</code>	Thêm phần tử X vào cuối hàng đợi Q
<code>dequeue(Q)</code>	Xóa phần tử tại đầu hàng đợi Q
<code>front(Q)</code>	Trả về phần tử đầu tiên của hàng đợi Q

www.ctu.edu.vn

90



CANTHO UNIVERSITY

HÀNG ĐỢI (QUEUE)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT HÀNG ĐỢI
 - DÙNG MẢNG DI CHUYỂN TỊNH TIẾN
 - DÙNG MẢNG VÒNG
 - DÙNG DSLK

91

www.ctu.edu.vn

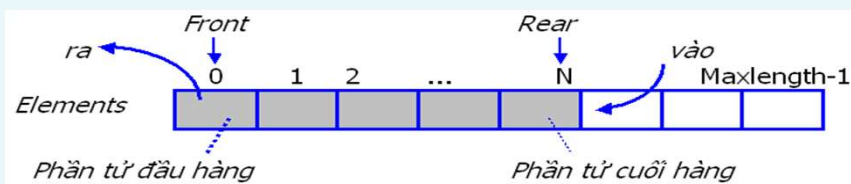


CANTHO UNIVERSITY

CÀI ĐẶT HÀNG BẰNG MẢNG DI CHUYỂN TỊNH TIẾN




- Mô hình



92

www.ctu.edu.vn




KHAI BÁO

CANTHO UNIVERSITY
93

```

#define MaxLength ...
    //chiều dài tối đa của mảng
typedef ... ElementType;
    //Kiểu dữ liệu của các phần tử trong hàng
typedef struct {
    ElementType Elements[MaxLength];
    //Lưu trữ nội dung các phần tử
    int Front, Rear;
    //chỉ số đầu và cuối hàng
} Queue;
Queue Q;
  
```

www.ctu.edu.vn



KHỞI TẠO HÀNG Q RỖNG

CANTHO UNIVERSITY
94

Front = -1
Rear = -1


	0	1	2	...		Maxlength-
<i>Elements</i>						

- Front và Rear không trở đến vị trí hợp lệ nào
- Ta cho front=rear=-1

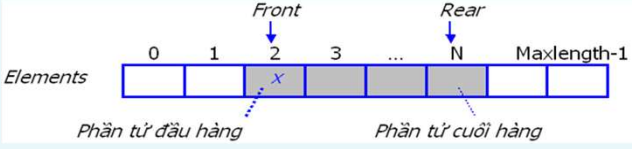
```

void makenullQueue(Queue *pQ) {
    pQ->Front=-1;
    pQ->Rear=-1;
}
  
```

www.ctu.edu.vn



TRẢ VỀ PHẦN TỬ ĐẦU HÀNG




=>Giải thuật: *Kết quả của phép toán trên là x*

- Nếu hàng Q rỗng thì thông báo lỗi
- Ngược lại, trả về giá trị được lưu trữ tại ô có chỉ số là Front

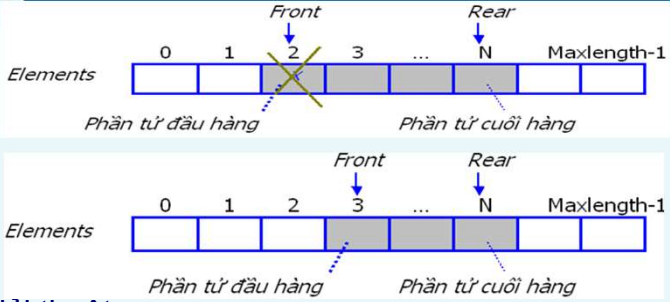
```

ElementType front(Queue Q) {
    if emptyQueue (Q)
        printf ("Hang rong");
    else
        return Q.Elements[Q.Front];
}
  
```

www.ctu.edu.vn
97



XÓA MỘT PHẦN TỬ KHỎI HÀNG(1)



=>Giải thuật:

- Nếu hàng Q rỗng thì thông báo lỗi
- Ngược lại:
 - Tăng Front lên 1 đơn vị
 - Nếu (Front > Rear) tức hàng chỉ còn 1 phần tử thì khởi tạo lại hàng rỗng luôn

www.ctu.edu.vn
98



CANTHO UNIVERSITY

XÓA MỘT PHẦN TỬ KHỎI HÀNG(2)

```
void deQueue (Queue *pQ) {
    if (!emptyQueue (*pQ)) {
        pQ->Front=pQ->Front+1;
        if (pQ->Front>pQ->Rear)
            makenullQueue(Q); //Dat lai hang rong
    }else printf("Loi: Hang rong!");
}
```

99

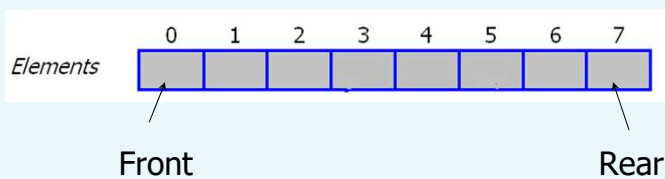
www.ctu.edu.vn



CANTHO UNIVERSITY

THÊM MỘT PHẦN TỬ VÀO HÀNG(1)

- Trường hợp hàng đầy



Thông báo lỗi "Lỗi: hàng đầy!"

100

www.ctu.edu.vn

THÊM MỘT PHẦN TỬ VÀO HÀNG(2)

- Trường hợp hàng chưa đầy nhưng bị tràn

Di chuyển tịnh tiến và gán lại Front và Rear

Rear = Rear + 1 và thêm phần tử vào vị trí Rear mới

101

www.ctu.edu.vn


THÊM MỘT PHẦN TỬ VÀO HÀNG(1)

- Trường hợp hàng chưa đầy và không bị tràn

Rear = Rear + 1 và thêm phần tử vào vị trí Rear mới

102

www.ctu.edu.vn




THÊM MỘT PHẦN TỬ VÀO HÀNG(3)

=>Giải thuật:

- Nếu hàng đầy thì thông báo lỗi
- Ngược lại, nếu hàng tràn thì phải tịnh tiến tất cả phần tử lên “Front-1” vị trí
- Tăng Rear 1 đơn vị và đưa giá trị x vào ô có chỉ số Rear mới này

103

www.ctu.edu.vn



THÊM MỘT PHẦN TỬ VÀO HÀNG(4)

```

void enqueue(ElementType X, Queue *pQ) {
    if (!fullQueue(*pQ)) {
        if (emptyQueue(*pQ)) pQ->Front=0;
        if (pQ->Rear==MaxLength-1) {
            //Di chuyển tịnh tiến ra trước Front -1 vị trí
            for(int i=pQ->Front; i<=pQ->Rear; i++)
                pQ->Elements[i-pQ->Front]=pQ->Elements[i];
            //Xác định vị trí Rear mới
            pQ->Rear=MaxLength - pQ->Front-1;
            pQ->Front=0;
        }
        //Tăng Rear để lưu nội dung mới
        pQ->Rear=pQ->Rear+1;
        pQ->Elements[pQ->Rear]=X;
    }
    else printf("Lỗi: Hàng đầy!");
}

```

104

www.ctu.edu.vn



CANTHO UNIVERSITY

HÀNG ĐỢI (QUEUE)

- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT HÀNG ĐỢI
 - DÙNG MẢNG DI CHUYỂN TỊNH TIẾN
 - DÙNG MẢNG VÒNG
 - DÙNG DSLK

www.ctu.edu.vn

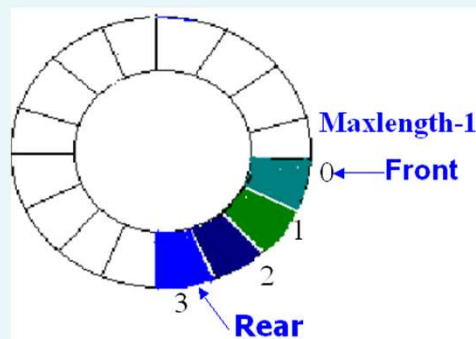
105



CANTHO UNIVERSITY


CÀI ĐẶT HÀNG BẰNG MẢNG VÒNG

- Mô hình



www.ctu.edu.vn

106

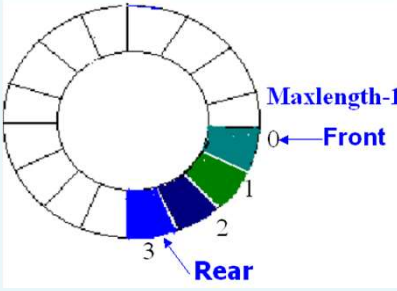


CÀI ĐẶT HÀNG BẰNG MẢNG VÒNG


- Mô hình
- Khai báo


```

#define MaxLength ...
typedef ... ElementType;
typedef struct {
    //Lưu trữ nội dung các phần tử
    ElementType Elements[MaxLength];
    //chỉ số đầu và đuôi hàng
    int Front, Rear;
} Queue;
      
```



www.ctu.edu.vn
107



KHỞI TẠO HÀNG RỖNG




- Front và Rear không trở đến vị trí hợp lệ nào
- Ta cho Front=Rear=-1

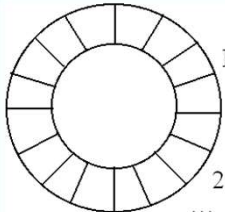
```

void makenullQueue (Queue *pQ) {
    pQ->Front=-1;
    pQ->Rear=-1;
}
      
```

www.ctu.edu.vn
108



KIỂM TRA HÀNG RỖNG



Maxlength-1
0
1
2
...


Hàng rỗng
Front → -1
Rear → -1

```

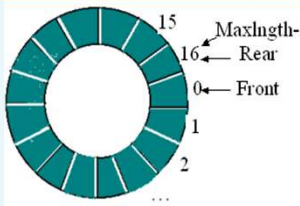
int emptyQueue (Queue Q) {
    return Q.Front == -1;
}

```

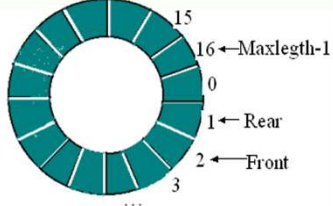
www.ctu.edu.vn
109



KIỂM TRA HÀNG ĐẦY



Maxlength-1
15 ← Rear
0 ← Front
1
2
...



Maxlength-1
15
16 ← Maxlength-1
0
1 ← Rear
2 ← Front
3
...

Trường hợp $Q.Rear = \text{Maxlength}-1$ và $Q.Front = 0$ Trường hợp $Q.Front = Q.Rear + 1$


- Hàng đầy khi số phần tử hiện có trong hàng bằng Maxlength

```

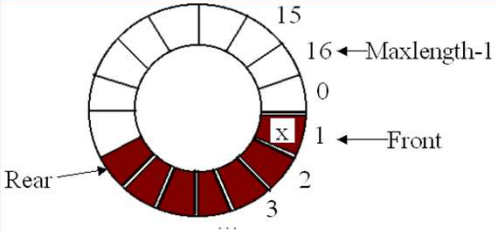
int fullQueue (Queue Q) {
    return (Q.Rear - Q.Front + 1) % MaxLength == 0;
}

```

www.ctu.edu.vn
110



LẤY GIÁ TRỊ PHẦN TỬ ĐẦU HÀNG




=>Giải thuật

- Nếu hàng Q rỗng thì thông báo lỗi
- Ngược lại, trả về giá trị được lưu trữ tại ô có chỉ số là Front

```

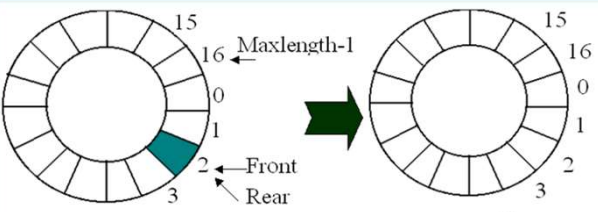
ElementType front(Queue Q){
    if (emptyQueue (Q))
        printf ("Hàng rỗng");
    else return Q.Elements[Q.Front];
}
  
```

www.ctu.edu.vn
111

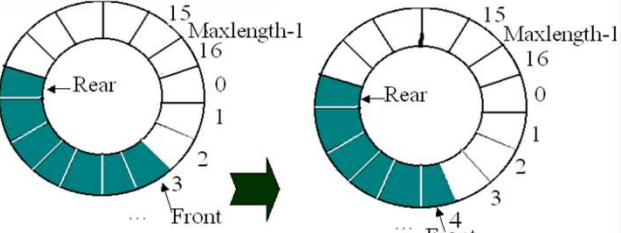


XÓA PHẦN TỬ ĐẦU HÀNG(1)

Các trường hợp có thể:



Hàng chỉ có 1 phần tử



Hàng có hơn 1 phần tử

tu.edu.vn
112



XÓA PHẦN TỬ ĐẦU HÀNG(2)

• Giải thuật :

- Nếu hàng Q rỗng thì thông báo lỗi
- Ngược lại:
 - Nếu $Front=Rear$ tức hàng chỉ còn 1 phần tử thì khởi tạo lại hàng rỗng
 - Ngược lại, thay đổi giá trị cho Front

```
void deQueue(Queue *pQ){
    if (!emptyQueue(*pQ)){
        if (pQ->Front==pQ->Rear)
            makenullQueue(pQ);
        else
            pQ->Front=(pQ->Front+1) % MaxLength;
    }else
        printf("Loi: Hang rong!");
}
```

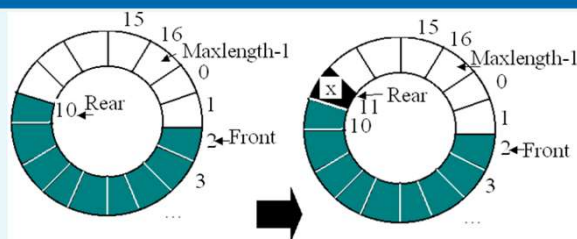
www.ctu.edu.vn

113

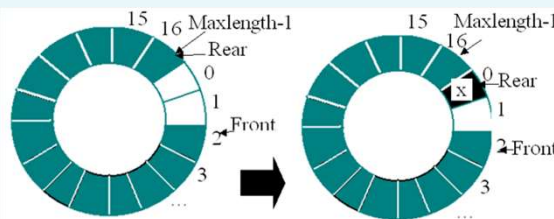


THÊM PHẦN TỬ X VÀO HÀNG Q(1)

- Các trường hợp có thể:



Trường hợp $Rear < Maxlength-1$



Trường hợp $Rear = Maxlength-1$

www.ctu.edu.vn

114



THÊM PHẦN TỬ X VÀO HÀNG Q(2)

- Giải thuật :
 - Nếu hàng đầy thì thông báo lỗi
 - Ngược lại, thay đổi giá trị Rear và đưa giá trị x vào ô có chỉ số Rear mới này

```
void enqueue(ElementType X, Queue *pQ) {
    if (!fullQueue(*pQ)) {
        if (emptyQueue(*pQ))
            pQ->Front=0;
        pQ->Rear=(pQ->Rear+1) % MaxLength;
        pQ->Elements[pQ->Rear]=X;
    } else printf("Loi: Hang day!");
}
```

115

www.ctu.edu.vn




HÀNG ĐỢI (QUEUE)

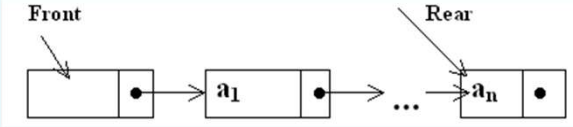
- ĐỊNH NGHĨA
- CÁC PHÉP TOÁN
- CÀI ĐẶT HÀNG ĐỢI
 - DÙNG MẢNG DI CHUYỂN TÍNH TIẾN
 - DÙNG MẢNG VÒNG
 - DÙNG DSLK

116

www.ctu.edu.vn



CÀI ĐẶT HÀNG BẰNG DSLK




– Dùng 2 con trỏ Front và Rear để chỉ tới phần tử đầu hàng và cuối hàng

- Khai báo

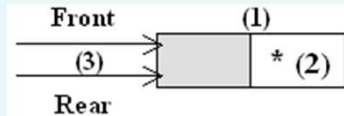
```

typedef ... ElementType; //kiểu phần tử của hàng
struct Node{
    ElementType Element;
    struct Node* Next;    //Con trỏ chỉ ô kế tiếp
};
typedef struct Node* Position;
typedef struct{
    Position Front, Rear; //2 con trỏ
} Queue;
Queue Q;
  
```

www.ctu.edu.vn
117



KHỞI TẠO HÀNG Q RỖNG




– Cho Front và rear cùng trỏ đến HEADER của hàng


```

void makenullQueue(Queue *pQ) {
    Position Header;
    Header=(struct Node*)malloc(sizeof(struct Node));
    //Cấp phát Header
    Header->Next=NULL;
    Q->Front=Header;
    Q->Rear=Header;
}
  
```

www.ctu.edu.vn
118



KIỂM TRA HÀNG Q RỖNG



KIỂM TRA HÀNG Q RỖNG


- Kiểm tra xem Front và Rear có cùng chỉ đến 1 ô (HEADER) không?

```


int emptyQueue (Queue Q) {
    return (Q.Front==Q.Rear);
}

```

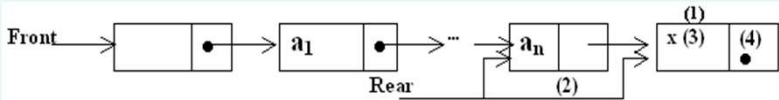
www.ctu.edu.vn
119



THÊM MỘT PHẦN TỬ X VÀO HÀNG Q



THÊM MỘT PHẦN TỬ X VÀO HÀNG Q



=>Giải thuật:


- Thêm 1 phần tử vào hàng ta thêm vào sau Rear (Rear->Next) 1 ô mới
- Cho Rear trở đến phần tử mới này và đặt giá trị thêm vào cho Rear
- Cho trường next của ô mới này trở tới NULL

```

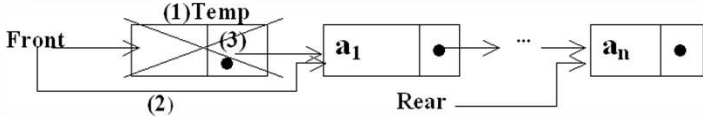
void enqueue(ElementType X, Queue *pQ) {
    pQ->Rear->Next=(struct Node*)malloc(sizeof(struct Node));
    // thêm 1 Phần tử vào sau Rear
    pQ->Rear=pQ->Rear->Next; // Trở Rear đến phần tử mới
    pQ->Rear->Element=X; //Đat gia tri them vao cho Rear
    pQ->Rear->Next=NULL; //Gán Next của Rear (ô mới) tới Null
}

```

www.ctu.edu.vn
120



XÓA MỘT PHẦN TỬ KHỎI HÀNG Q




- Để xóa 1 phần tử khỏi hàng ta chỉ cần cho Front trở tới vị trí kế tiếp của nó trong danh sách

```

void deQueue (Queue *pQ) {
    if (!emptyQueue (pQ)) {
        Position Tempt;
        Tempt=pQ->Front;
        pQ->Front=pQ->Front->Next;
        free (Tempt);
    } else printf("Loi : Hang rong");
}
  
```


www.ctu.edu.vn
121



CÁC ỨNG DỤNG CỦA NGĂN XẾP VÀ HÀNG ĐỢI

- Bạn hãy liệt kê một số ứng dụng có sử dụng
 - Ngăn xếp
 - Hàng đợi

www.ctu.edu.vn
122




CÁC ỨNG DỤNG CỦA NGĂN XẾP VÀ HÀNG ĐỢI

- Ngăn xếp
 - Khử hàm đệ quy (Recursive Function)
 - Đánh giá biểu thức và phân tích cú pháp (Expression evaluation and syntax parsing)
 - Gọi hàm (Calling Function)
 - Thuật toán quay lui (Backtracking) vd: Puzzle, Sudoku...
- Hàng đợi
 - Quản lý in trên mạng.
 - Quản lý truyền thông điệp giữa 02 tiến trình, chương trình hay hệ thống.
 - Lập lịch biểu (VD: lịch cất cánh hay đáp máy bay trên 1 đường băng)

123

www.ctu.edu.vn



BÀI TẬP

Bài 1:


- Viết hàm để in các phần tử trong ngăn xếp.
- Viết hàm để in các phần tử trong hàng đợi.

Bài 2:

- Viết chương trình nhập vào một ngăn xếp chứa các số nguyên
- Sau đó sử dụng một hàng đợi để đảo ngược thứ tự của các phần tử trong ngăn xếp đó

124


www.ctu.edu.vn



Trả lời bài tập

CANTHO UNIVERSITY

www.ctu.edu.vn 125




NỘI DUNG SẼ HỌC

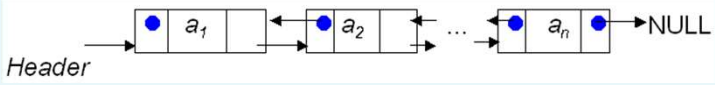
CANTHO UNIVERSITY

- Kiểu dữ liệu trừu tượng danh sách (LIST)
- Kiểu dữ liệu trừu tượng ngăn xếp (STACK)
- Kiểu dữ liệu trừu tượng hàng đợi (QUEUE)
- Danh sách liên kết kép (Doubly-Linked Lists)

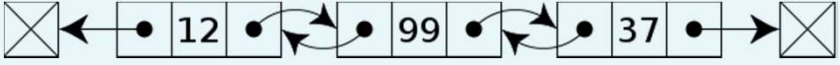
www.ctu.edu.vn 126



DANH SÁCH LIÊN KẾT KÉP




- Một phần tử của danh sách gồm 3 phần:
 - Element để lưu giữ nội dung phần tử
 - Next để chỉ đến phần tử đứng sau phần tử đang xét
 - Previous để chỉ đến phần tử đứng trước phần tử đang xét
- Để quản lý danh sách liên kết kép ta dùng một con trỏ được gọi là Header
 - Header có cùng kiểu với kiểu phần tử trong danh sách
 - Header có thể được cấp phát bộ nhớ hay không cấp phát bộ nhớ

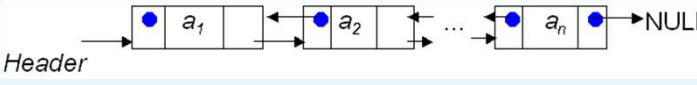


www.ctu.edu.vn

127



DANH SÁCH LIÊN KẾT KÉP




- Khai báo


```

typedef ... ElementType; //kiểu nội dung của phần tử
struct Node{
    ElementType Element;
    //lưu trữ nội dung phần tử
    struct Node* Prev;
    struct Node* Next;
    //Con trỏ trỏ tới phần tử trước và sau
};
typedef struct Node* Position;
typedef Position DoubleList;
```

www.ctu.edu.vn

128



CANTHO UNIVERSITY

ƯU ĐIỂM CỦA DSLK KÉP

- Theo bạn, thuận lợi và bất lợi của việc sử dụng danh sách liên kết kép là gì?

www.ctu.edu.vn 129



CANTHO UNIVERSITY

Hết chương

www.ctu.edu.vn 130