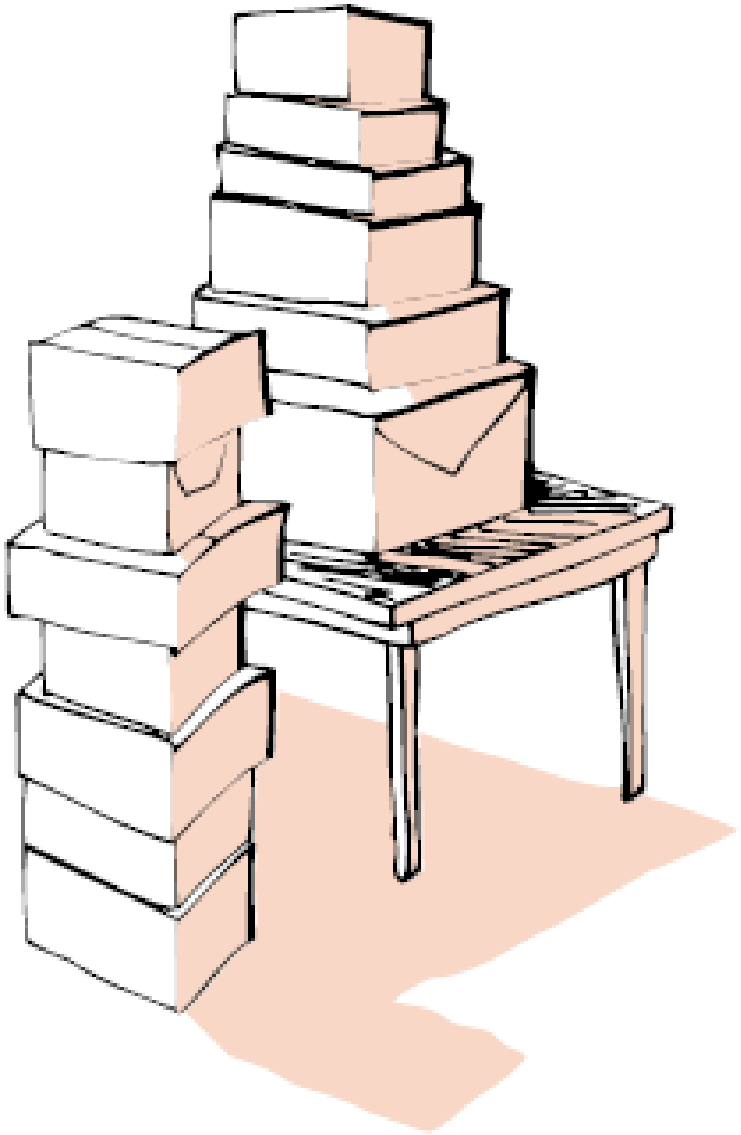
NỘI DUNG

Stack

ThS.Nguyễn Thúy Loan

Mô tả stack

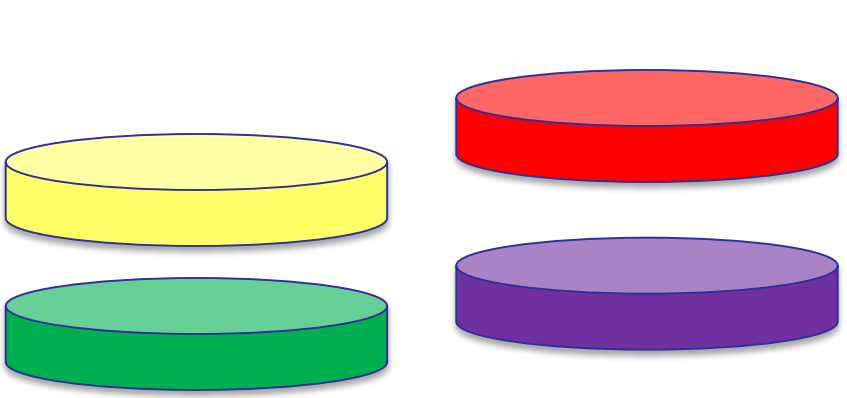


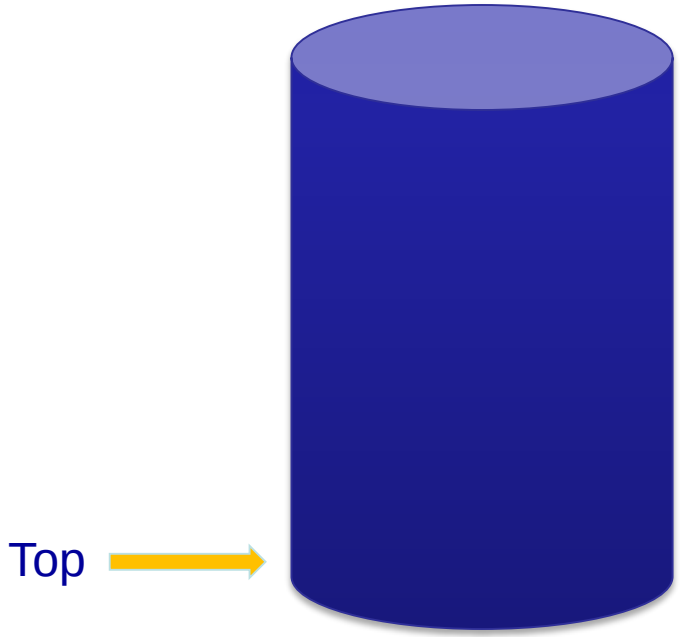
- Một stack là một cấu trúc dữ liệu mà việc thêm vào và loại bỏ được thực hiện tại một đầu (gọi là đỉnh – top của stack).
- Là một dạng vào sau ra trước – LIFO (Last In First Out)

Mô tả stack

I. NGĂN XẾP (STACK)

1. Giới Thiệu





LIFO: Last In First Out - Vào Sau Ra Trước.

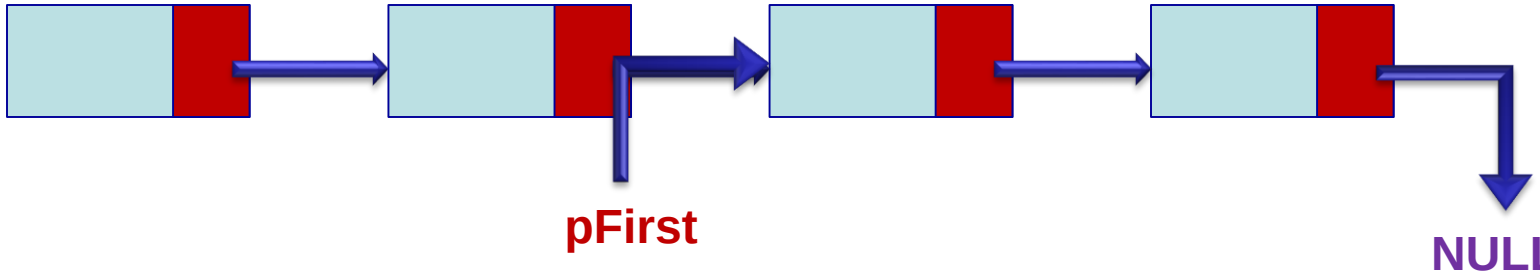
I. NGĂN XẾP (STACK)

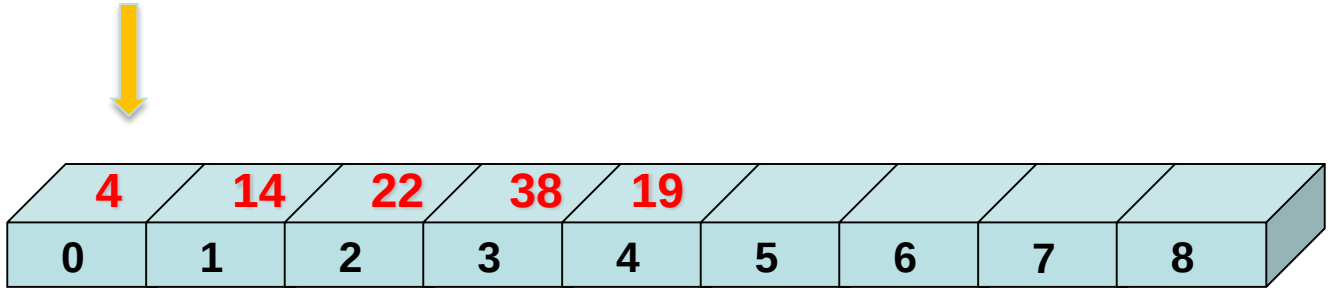
1. Giới Thiệu

❖ Ngăn xếp (Stack) là một danh sách mà ta giới hạn việc thêm vào hoặc loại bỏ một phần tử chỉ thực hiện tại một đầu của danh sách, đầu này gọi là đỉnh (TOP) của ngăn xếp.

❖ **LIFO**: Last In First Out - vào sau ra trước.

2. Khai báo cấu trúc dữ liệu cho stack





KHAI BÁO CẤU TRÚC DỮ LIỆU CHO STACK

Khai báo ngăn xếp dạng dãy

```
# define size 100
struct Stack
{
      int n;
      <Kiểu dữ liệu> <Tên mảng> [size ];
}
```

Khai báo ngăn xếp dạng DSLK

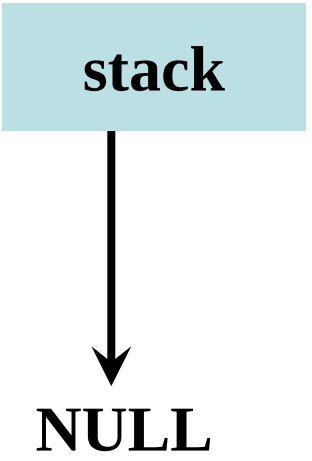
```
typedef struct node
{
            int info;
            struct node * pNext;
};

node *stack ;
```

1. KHỞI TẠO

Khái niệm: Khởi tạo ngăn xếp (stack) là tạo ra ngăn xếp rỗng không chứa đối tượng nào hết.

- Định nghĩa hàm

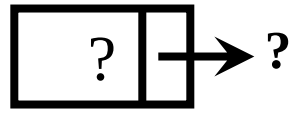
```
1.void Init(node *&stack)
2.{
3.     stack = NULL;
4.}
```

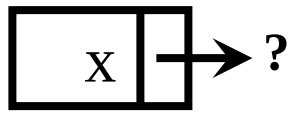


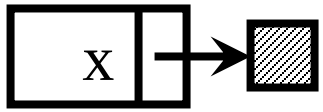
5. TẠO NODE CHO STACK

- **Khái niệm: Tạo node cho STACK là xin cấp phát bộ nhớ có kích thước bằng với kích thước của kiểu dữ liệu node để chứa thông tin đã được biết trước.**

```
1.node* GetNode(int x)
2.{    node *p=new NODE;
3.     if (p==NULL)  return NULL;
4.      p->info = x;
5.      p->pNext = NULL;
6.     return p;
7.}
```

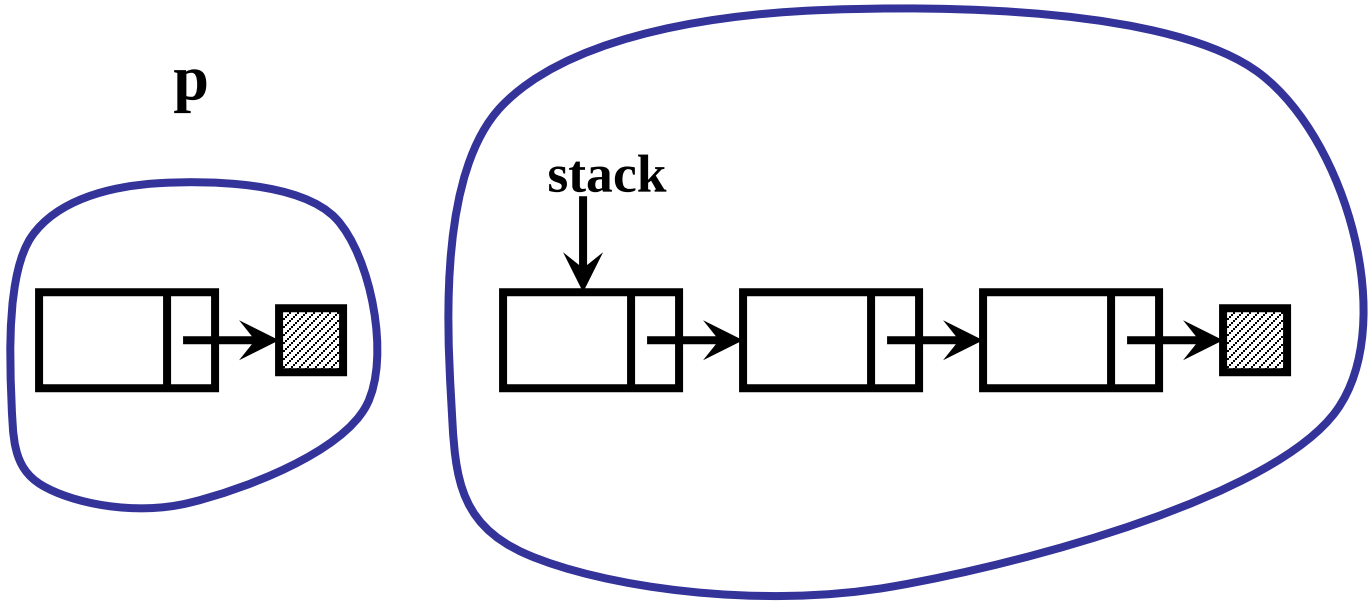






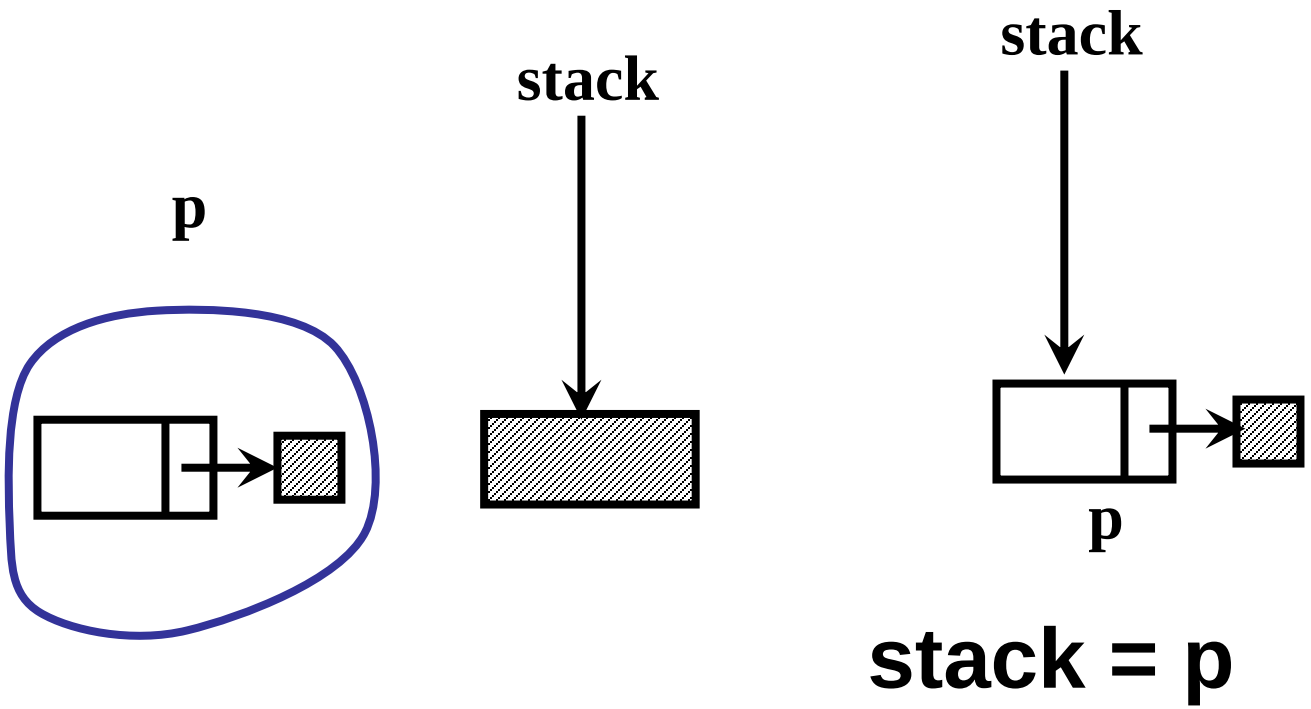
6. THÊM MỘT NODE VÀO NGĂN XẾP

- Khái niệm: Thêm một node vào NGĂN XẾP là gắn node đó vào đầu NGĂN XẾP.
- Hình vẽ



6. THÊM MỘT NODE VÀO NGĂN XẾP

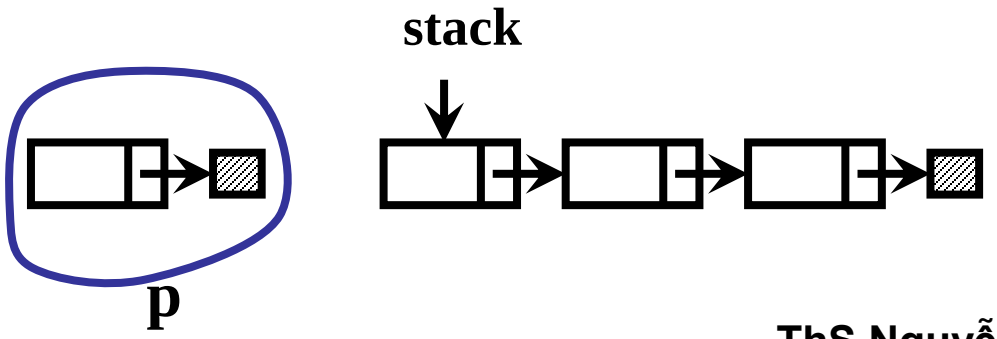
- Khái niệm: Thêm một node vào STACK là gắn node đó vào đầu STACK.



6. THÊM MỘT NODE VÀO STACK

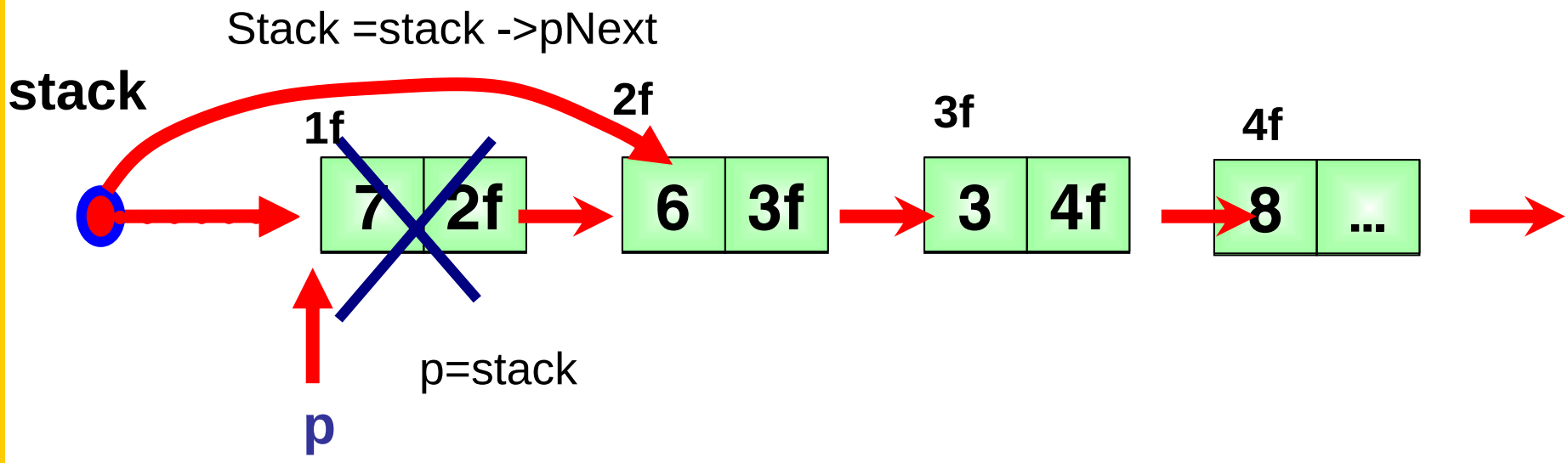
- Định nghĩa hàm:

```
1.void AddHead(node *&stack, node*p)
2.{
3.     if( stack==NULL)
4.          stack= p;
5.     else
6.     {
7.          p->pNext = stack;
8.          stack = p;
9.     }
10.}
```



Thuật toán hủy phần tử trong STACK

- Khái niệm: HỦY một node trong STACK là HỦY node đầu STACK.



Thuật toán hủy phần tử đầu trong DSLK

```
void  deletehead ( node *& stack )
{
        node *p;
        if ( stack !=NULL )
        {
                p= stack ;
                stack  = stack ->pnext;
                delete(p);
        }
}
```

Ví dụ: Nhập danh sách liên kết đơn các số nguyên.

```
1.void Input(node *&stack)
2.{    int x;
3.     Init(l );
4.     do{
5.          printf("Nhap gia tri x: ");
6.          scanf("%d",&x);
7.          if( x!=0)
8.          {
9.             NODE*p=GetNode(x);
10.            if (p!=NULL)
11.               Addhead(l,p);
12.         }
13.       }while(x!=0)
14.}
```

Ví dụ: Đổi thập phân sang nhị phân.

```
1.void Input(node *&stack, int n)
2.{    int x;
3.     Init(stack );
4.     do{
5.          x=n%2;
            n=n/2;
6.          NODE*p=GetNode(x);
7.             if (p!=NULL)
8.                 Addhead(stack,p);
                else
                {

9.              }
10.        }while(n!=0)
11.}
```

8. DUYỆT TUẦN TỰ DANH SÁCH LIÊN KẾT ĐƠN

```
1.void xuatvaxoastack (NODE *&stack)
2.{    int a[16], i=0;
3.     while(stack!=NULL)
4.     {
          a[i]= stack->info ;
          i++;
5.         deletehead( stack );
6.     }
7.     for( int j=0; j<=i; j++ )
          print( "%d", a[i] );
8.}
```

Khai báo CTDL cho stack kiểu mảng

```
1.struct stack
2.{
3.      int n;
4.      KDL a[100];

5.};
6.typedef struct stack STACK;
```

- KDL là kiểu dữ liệu của đối tượng được lưu trong stack.
- Cài đặt bằng mảng

Khai báo CTDL cho stack kiểu mảng

```
1.struct stack
2.{
3.      int n;
4.      KDL a[100];

5.};typedef struct stack STACK;
```

STACK st ;

st.n

st. a[st.n]

KHỞI TẠO

- Khái niệm: Khởi tạo ngăn xếp là tạo ra ngăn xếp rỗng không chứa đối tượng nào hết.

- Định nghĩa hàm

```
1.void Init(STACK &st)
2.{
3.     st.n=0;
4.}
```

KIỂM TRA NGĂN XẾP ĐẦY

- Khái niệm: Kiểm tra ngăn xếp đầy là hàm trả về giá trị 1 khi ngăn xếp đã đầy. Trong trường hợp ngăn xếp chưa đầy thì hàm trả về giá trị 0.

- Định nghĩa hàm:

```
1.int IsFull(STACK st)
2.{
3.     if(st.n==100)
4.          return 1;
5.     return 0;
6.}
```

THÊM MỘT ĐỐI TƯỢNG VÀO NGĂN XẾP

- Khái niệm: Thêm một đối tượng vào trong ngăn xếp xét về mặt kỹ thuật với CTDL đã được khai báo bên trên là việc thêm đối tượng đó vào cuối mảng a đang có n phần tử của stack mà thôi.
- Định nghĩa hàm

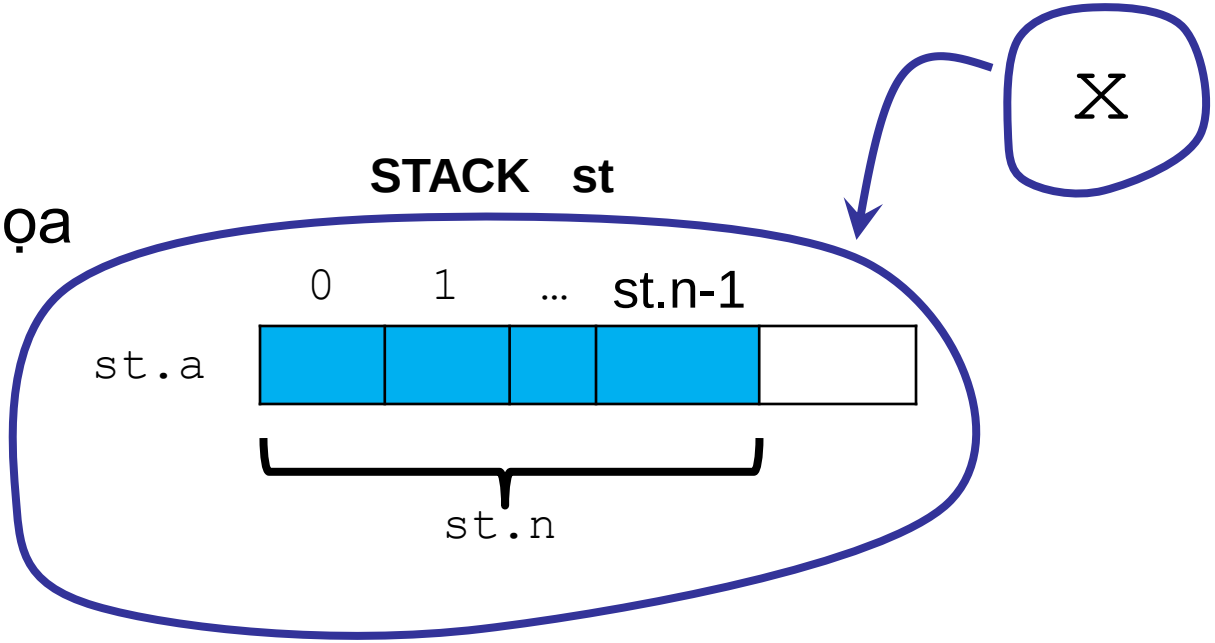
```
1.void Push(STACK &st, KDL x)
2.{
3.      st.a[st.n] = x;
4.      st.n++;
5.}
```

THÊM MỘT ĐỐI TƯỢNG VÀO NGĂN XẾP

Định nghĩa hàm

```
1. void Push(STACK &st, KDL x)
2. {
3.     st.a[st.n] = x;
4.     st.n++;
5. }
```

Hình vẽ minh họa



LẤY MỘT ĐỐI TƯỢNG RA KHỎI NGĂN XẾP

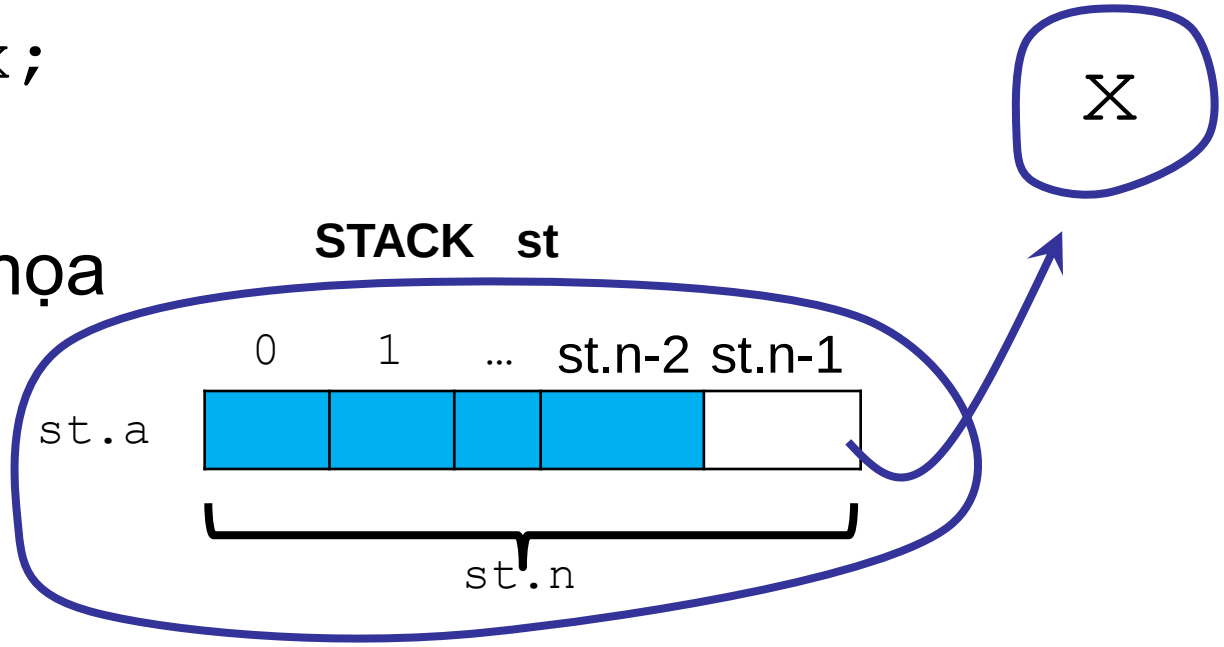
- Khái niệm: Lấy một đối tượng ra khỏi ngăn xếp xét về mặt kỹ thuật với CTDL đã được khai báo bên trên là việc lấy đối tượng cuối mảng a của stack ra khỏi mảng mà thôi.
- Định nghĩa hàm

```
1.KDL Pop(STACK &st)
2.{
3.    KDL x = st.a[st.n-1];
4.    st.n--;
5.    return x;
6.}
```

LẤY MỘT ĐỐI TƯỢNG RA KHỎI NGĂN XẾP

```
1.KDL Pop(STACK &st)
2.{
3.    KDL x = st.a[st.n-1];
4.    st.n--;
5.    return x;
6.}
```

7. Hình vẽ minh họa



QUEUE – HÀNG ĐỢI

MÔ TẢ QUEUE- HÀNG ĐỢI

- Một queue là một cấu trúc dữ liệu mà việc thêm vào được thực hiện ở một đầu (rear) và việc lấy ra được thực hiện ở đầu còn lại (front)
- Phần tử vào trước sẽ ra trước – FIFO (First In First Out)

KHAI BÁO CTDL CHO HÀNG ĐỢI

Khai báo hàng đợi bằng mảng

```
# define size 100
typedef    struct Queue
{
      int n;
      <Kiểu dữ liệu> <Tên mảng> [size ];
} ;
```

Khai báo hàng đợi dạng DSLK

```
typedef  struct node
{
            int info;
            struct   node* pNext;
};
 typedef    struct Queue
{
            node* Front;
            node* Rear;
};
```

CTDL CHO QUEUE- HÀNG ĐỢI

```
1.struct queue
2.{
3.      int n;
4.      KDL a[100];
5.};
6.typedef struct queue QUEUE;
```

- KDL là kiểu dữ liệu của đối tượng được lưu trong `queue`.
- Cài đặt bằng mảng

KHỞI TẠO QUEUE- HÀNG ĐỢI

- Khái niệm: Khởi tạo hàng đợi là tạo ra hàng đợi rỗng không chứa đối tượng nào hết.

- Định nghĩa hàm

```
1.void Init(QUEUE &que)
2.{
3.        que.n=0;
4.}
```

KIỂM TRA HÀNG ĐỢI RỖNG

- Khái niệm: Kiểm tra hàng đợi rỗng là hàm trả về giá trị 1 khi hàng đợi rỗng. Trong tình huống hàng đợi chưa rỗng thì hàm sẽ trả về giá trị 0.
- Định nghĩa hàm

```
1.int IsEmpty(QUEUE que)
2.{
3.     if (que.n==0)
4.          return 1;
5.     return 0;
6.}
```

KIỂM TRA HÀNG ĐỢI ĐẦY

- Khái niệm: Kiểm tra hàng đợi đầy là hàm trả về giá trị 1 khi hàng đợi đã đầy và trả về giá trị 0 khi hàng đợi chưa đầy.
- Định nghĩa hàm:

```
1.int IsFull(QUEUE que)
2.{
3.     if(que.n==100)
4.          return 1;
5.     return 0;
6.}
```

THÊM MỘT ĐỐI TƯỢNG VÀO QUEUE

- Khái niệm: Thêm một đối tượng vào hàng đợi xét về mặt kỹ thuật với CTDL đã được khai báo bên trên là việc thêm đối tượng đó vào cuối mảng a đang có n phần tử của hàng đợi mà thôi.
- Định nghĩa hàm

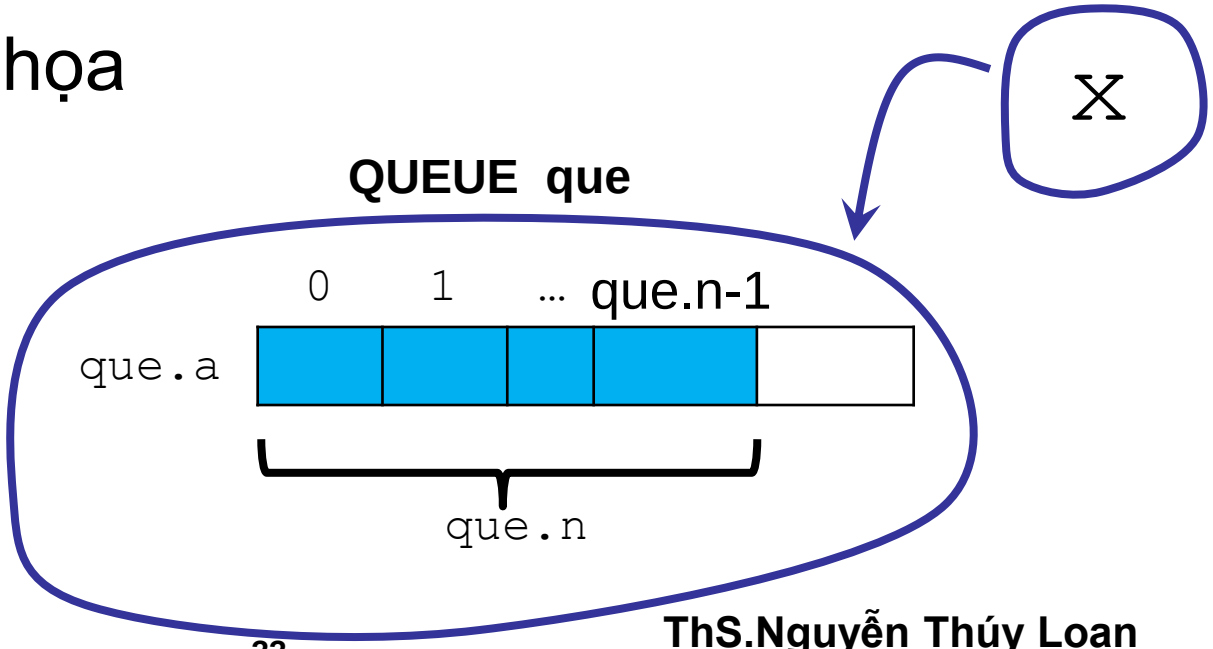
```
1.void EnQueue(QUEUE &que,KDL x)
2.{
3.      que.a[que.n] = x;
4.      que.n++;
5.}
```

QUEUE- HÀNG ĐỢI

- Định nghĩa hàm

```
1.void EnQueue(QUEUE &que, KDL x)
2.{
3.      que.a[que.n] = x;
4.      que.n++;
5.}
```

- Hình vẽ minh họa

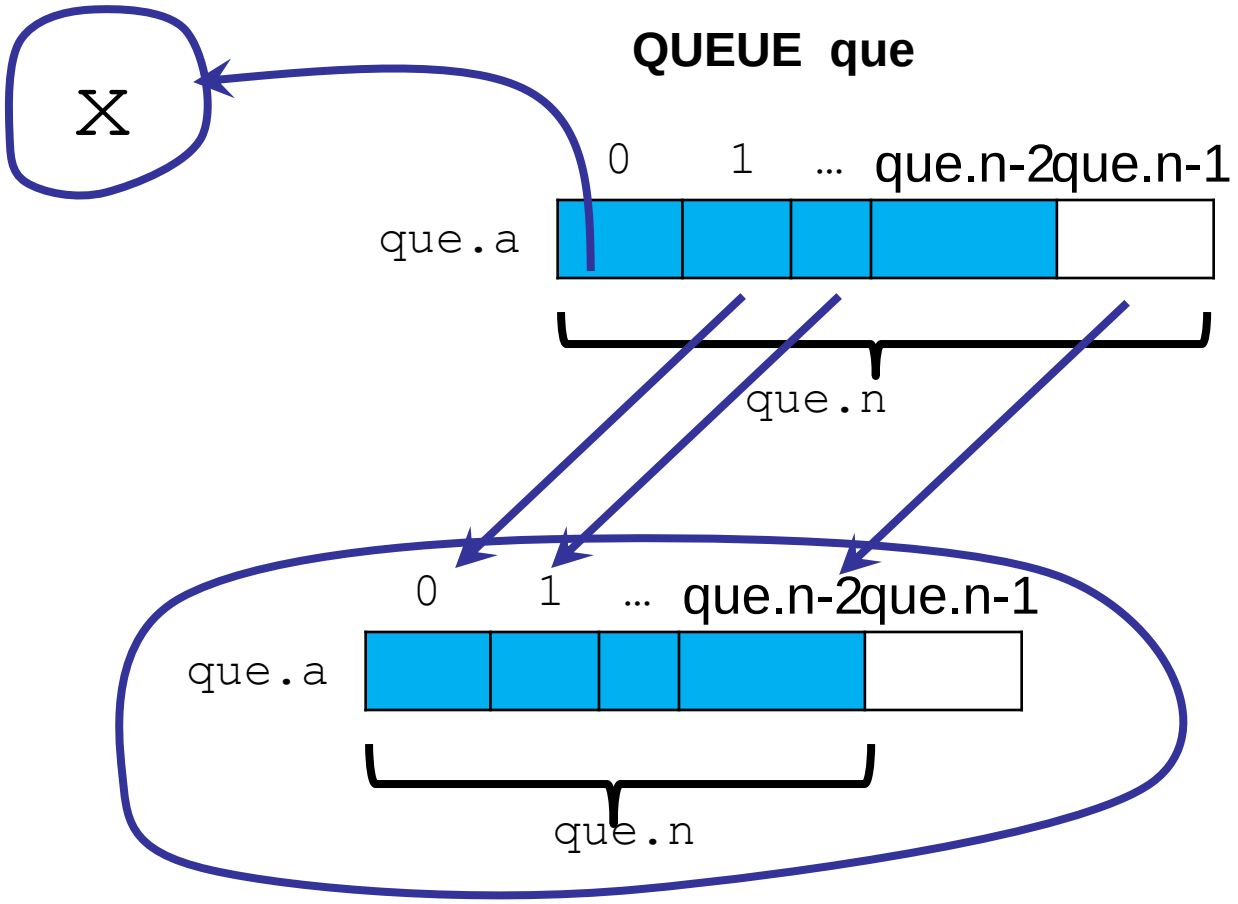


LẤY MỘT ĐỐI TƯỢNG RA KHỎI QUEUE

- Khái niệm: Lấy một đối tượng ra khỏi hàng đợi xét về mặt kỹ thuật với CTDL đã được khai báo bên trên là việc lấy đối tượng đầu mảng a của hàng đợi (queue) ra khỏi mảng mà thôi.
- Định nghĩa hàm

```
1. KDL    DeQueue(QUEUE &que)
2. {
3.     KDL x = que.a[0];
4.     for(int i=0 ; i< = que.n-2 ; i++ )
5.         que.a[i] = que.a[i+1];
6.     que.n--;
7.     return x;
8. }
```

QUEUE- HÀNG ĐỢI



LẤY MỘT ĐỐI TƯỢNG RA KHỎI QUEUE

```
1.KDL    DeQueue(QUEUE &que)
2.{
3.    KDL x = que.a[0];
4.    for(int i=0 ; i< = que.n-2 ; i++ )
5.        que.a[i] = que.a[i+1];
6.    que.n--;
7.    return x;
8.}
```