



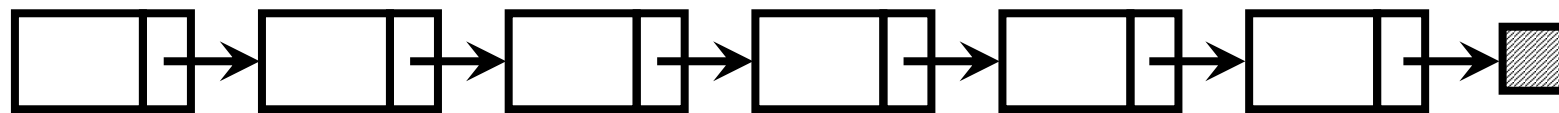
DANH SÁCH LIÊN KẾT ĐƠN

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



HÌNH ẢNH DANH SÁCH LIÊN KẾT ĐƠN

Hình ảnh danh sách liên kết đơn



NULL

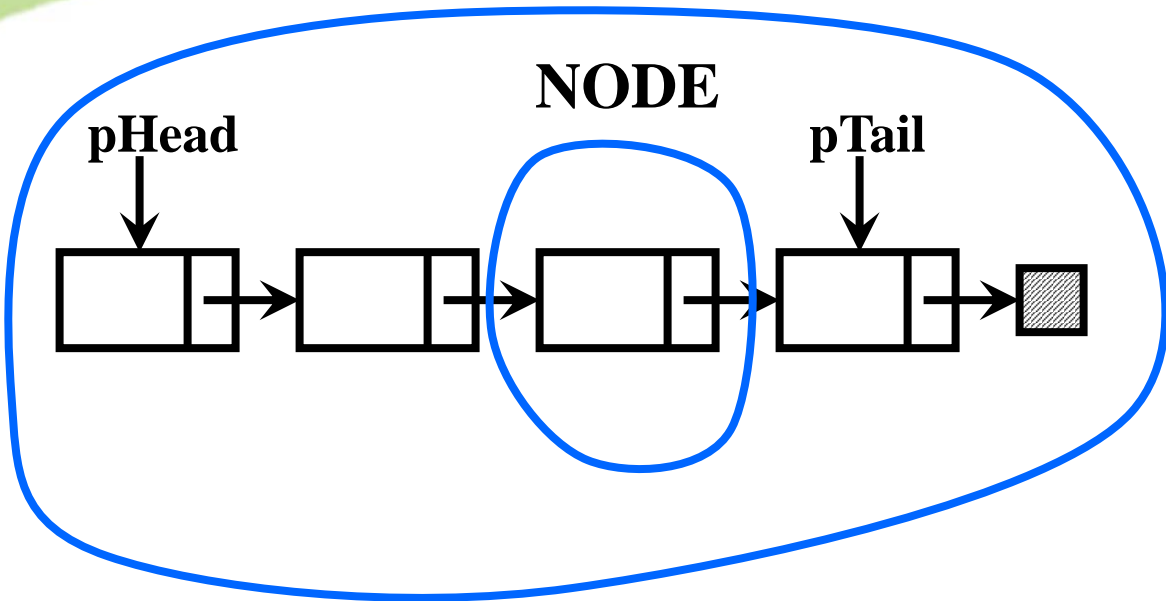
Hình ảnh danh sách
liên kết đơn



CẤU TRÚC DỮ LIỆU DSLK ĐƠN

LIST

NODE



Cấu trúc dữ liệu danh sách liên kết đơn

```
11.struct node
```

```
12.{
```

```
13.|    KDL info;
```

```
14.|    struct node* pNext;
```

```
15.};
```

```
16.typedef struct node NODE;
```

```
17.struct list
```

```
18.{
```

```
19.|    NODE* pHead;
```

```
20.|    NODE* pTail;
```

```
21.};
```

```
22.typedef struct list LIST;
```

CTDL DSLK Đơn



```
11.struct node
12.{
13.|    KDL info;
14.|    struct node* pNext;
15.};
16.typedef struct node NODE;
17.struct list
18.{
19.|    NODE* pHead;
20.|    NODE* pTail;
21.};
22.typedef struct list LIST;
```

— Câu lệnh 14 có ba ý:

- + **pNext** là con trỏ kiểu cấu trúc **NODE**.
- + Miền giá trị của thành phần **pNext** là địa chỉ ô nhớ
- + Về mặt bản chất thành phần **pNext** giữ địa chỉ của **NODE** kế tiếp trong danh sách liên kết đơn.

CTDL DSLK Đơn



```
11.struct node
12.{
13.|    KDL info;
14.|    struct node* pNext;
15.};
16.typedef struct node NODE;
17.struct list
18.{
19.|    NODE* pHead;
20.|    NODE* pTail;
21.};
22.typedef struct list LIST;
```

— Câu lệnh 19 có ba ý:

- + **pHead** là con trỏ kiểu cấu trúc **NODE**.
- + Miền giá trị của thành phần **pHead** là địa chỉ ô nhớ
- + Về mặt bản chất thành phần **pHead** giữ địa chỉ của **NODE** đầu tiên trong danh sách liên kết đơn.

CTDL DSLK Đơn



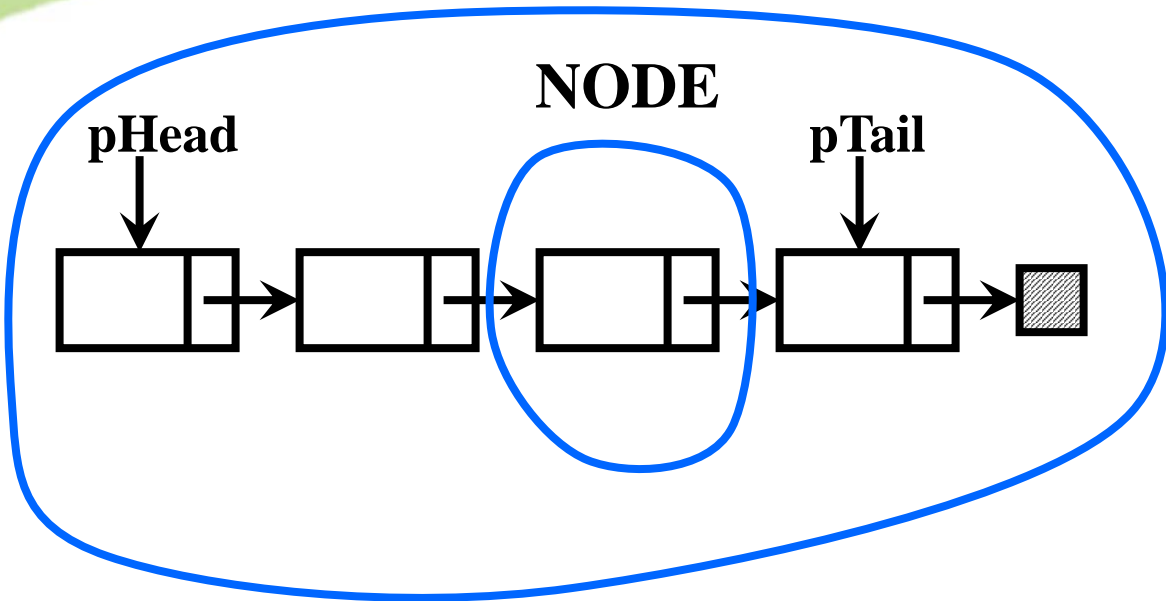
```
11.struct node
12.{
13.|    KDL info;
14.|    struct node* pNext;
15.};
16.typedef struct node NODE;
17.struct list
18.{
19.|    NODE* pHead;
20.|    NODE* pTail;
21.};
22.typedef struct list LIST;
```

— Câu lệnh 20 có ba ý:

- + **pTail** là con trỏ kiểu cấu trúc **NODE**.
- + Miền giá trị của thành phần **pTail** là địa chỉ ô nhớ
- + Về mặt bản chất thành phần **pTail** giữ địa chỉ của **NODE** cuối cùng trong danh sách liên kết đơn.

LIST

NODE



CTDL dslk đơn các
số nguyên

```
11.struct node
```

```
12.{
```

```
13.|    int info;
```

```
14.|    struct node* pNext;
```

```
15.};
```

```
16.typedef struct node NODE;
```

```
17.struct list
```

```
18.{
```

```
19.|    NODE* pHead;
```

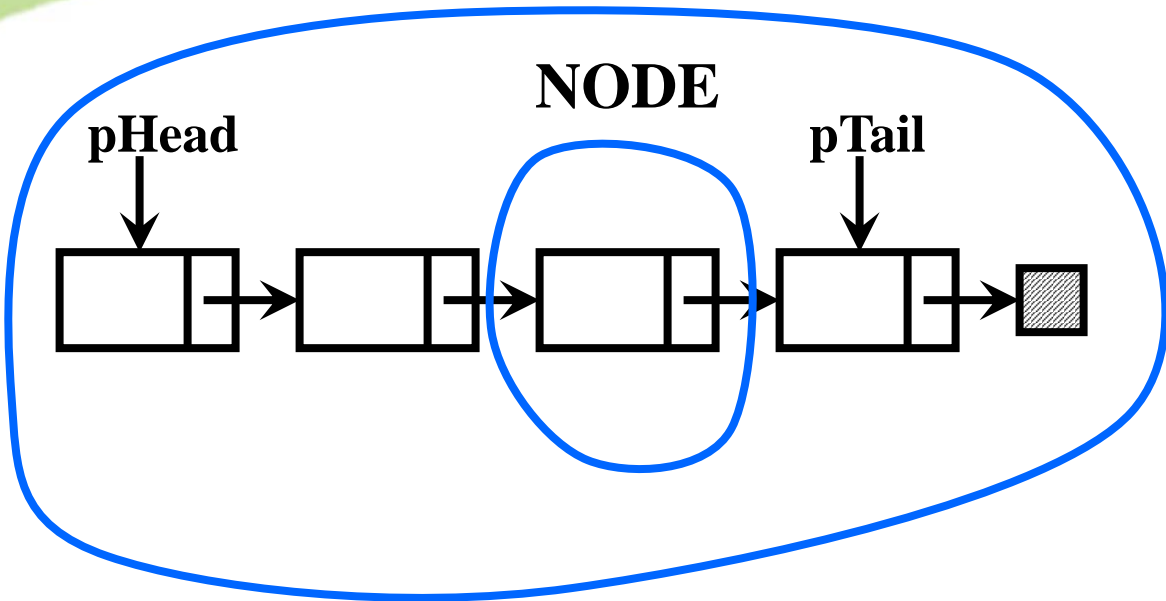
```
20.|    NODE* pTail;
```

```
21.};
```

```
22.typedef struct list LIST;
```

LIST

NODE



CTDL dslk đơn các
số thực

```
11.struct node
```

```
12.{
```

```
13.|    float info;
```

```
14.|    struct node* pNext;
```

```
15.};
```

```
16.typedef struct node NODE;
```

```
17.struct list
```

```
18.{
```

```
19.|    NODE* pHead;
```

```
20.|    NODE* pTail;
```

```
21.};
```

```
22.typedef struct list LIST;
```



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 02

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



KHỞI TẠO DANH SÁCH LIÊN KẾT ĐƠN

Khởi tạo danh sách liên kết đơn



— Khái niệm: Khởi tạo danh sách liên kết đơn là tạo ra danh sách rỗng không chứa node nào hết.

— Khai báo hàm:

```
11. void Init(LIST&);
```

— Định nghĩa hàm khởi tạo danh sách liên kết đơn.

```
11. void Init(LIST& l)
12. {
13.     l.pHead = NULL;
14.     l.pTail = NULL;
15. }
```

Khởi tạo danh sách
liên kết đơn



KIỂM TRA DSLK ĐƠN RỒI



Kiểm tra dslk đơn rỗng

- Khái niệm: Kiểm tra danh sách liên kết đơn rỗng là hàm trả về giá trị 1 khi danh sách rỗng. Trong tình huống danh sách không rỗng thì hàm sẽ trả về giá trị 0.

Kiểm tra danh sách
liên kết đơn rỗng

- Khai báo hàm:

```
11. int IsEmpty(LIST);
```

- Định nghĩa hàm kiểm tra danh sách liên kết đơn rỗng.

```
11. int IsEmpty(LIST l)  
12. {  
13.     if(l.pHead==NULL)  
14.         return 1;  
15.     return 0;  
16. }
```




Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 03

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



TẠO NODE CHO DSLK ĐƠN



Tạo node cho dslk đơn

— Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

— Khai báo hàm:

```
11. NODE* GetNode(KDL);
```

— Các câu hỏi:

- + Tên hàm là gì?
- + Kiểu dữ liệu trả về là gì?
- + Có mấy tham số đầu vào?
- + Kiểu dữ liệu của tham số đầu vào là gì?
- + Loại tham số đầu vào là gì?



Tạo node cho dslk đơn

- **Khái niệm:** Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh
sách liên kết đơn

```
11. NODE* GetNode(KDL x)
12. {
    |
    |
    |
19. }
```



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

Tên hàm tạo node cho dslk đơn là `GetNode`.

```
19. }
```



Tạo node cho dslk đơn

- **Khái niệm:** Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

Có một tham số đầu vào, tên tham số là x, tham số là tham trị.

```
19. }
```



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

Kiểu dữ liệu trả về của hàm `GetNode` là con trỏ kiểu cấu trúc `NODE`.

```
19. }
```




Tạo node cho dslk đơn

- **Khái niệm:** Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

Về mặt bản chất
hàm **GetNode** sẽ trả
về một địa chỉ ô nhớ.

```
19. }
```



Tạo node cho dslk đơn

- **Khái niệm:** Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh
sách liên kết đơn

```
11. NODE* GetNode(KDL x)
12. {
    NODE* p=new NODE;
    19. }
```



Tạo node cho dslk đơn

- **Khái niệm:** Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

```
    NODE* p=new NODE;
```

p là một biến con trỏ kiểu cấu trúc **NODE**.

```
19. }
```



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

```
    NODE* p=new NODE;
```

Miền giá trị của biến con trỏ `p` là địa chỉ ô nhớ.

```
19. }
```



Tạo node cho dslk đơn

- **Khái niệm:** Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh
sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

```
    NODE* p=new NODE;
```

`new NODE` là xin cấp phát động một vùng nhớ có kích thước bằng kích thước của kiểu dữ liệu `NODE`.

```
19. }
```



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

```
    NODE* p=new NODE;
```

Nếu việc cấp phát thành công OS sẽ trả về địa chỉ ô nhớ đầu tiên của vùng nhớ được cấp phát, địa chỉ ô nhớ này được gán cho biến con trỏ `p`.

```
19. }
```



Tạo node cho dslk đơn

- **Khái niệm:** Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

```
    NODE* p=new NODE;
```

Nếu việc cấp phát thất bại, OS sẽ trả về một địa chỉ đặc biệt là địa chỉ NULL, địa chỉ NULL này sẽ được gán cho biến con trỏ **p**.

```
19. }
```



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
```

```
12. {
```

```
    NODE* p=new NODE;
```

```
19. }
```

Như vậy, thông thường sau câu lệnh thứ 13, biến con trỏ **p** sẽ giữ địa chỉ ô nhớ đầu tiên của vùng nhớ có kích thước bằng kích thước của KDL **NODE**.



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
12. {
13.     NODE* p=new NODE;
14.     if(p==NULL)
15.         return NULL;
16.     p->info = x;
17.     p->pNext=NULL;
18.     return p;
19. }
```



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
12. {
13. |
14. |
15. |
16. |
17. |
18. | return p;
19. }
```

Kết thúc lời gọi hàm và trả về địa chỉ ô nhớ đang được lưu trong biến con trỏ `p`.

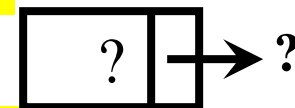


Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
12. {
13.     NODE* p=new NODE;
14.     if(p==NULL)
15.         return NULL;
16.     p->info = x;
17.     p->pNext=NULL;
18.     return p;
19. }
```





Tạo node cho dslk đơn

- Phân tích câu lệnh dòng 11

11. **NODE* GetNode(KDL x)**

- Tên hàm tạo node cho dslk đơn là **GetNode**.

- Có một tham số đầu vào, tên tham số là x, tham số là tham trị.

- KDL trả về của hàm **GetNode** là con trỏ kiểu cấu trúc **NODE**.

- Về mặt bản chất hàm **GetNode** sẽ trả về một địa chỉ ô nhớ.

- Phân tích câu lệnh dòng 18

18. **return p;**

- Kết thúc lời gọi hàm và trả về địa chỉ ô nhớ đang được lưu trong biến con trỏ **p**.

- Ý nghĩa của địa chỉ ô nhớ đang lưu biến con trỏ **p** xem ở slide ngay sau.

Tạo node cho dslk đơn



Phân tích dòng lệnh 13.

13. `NODE *p=new NODE;`

- `p` là một biến con trỏ kiểu cấu trúc `NODE`.
- Miền giá trị của biến con trỏ `p` là địa chỉ ô nhớ.
- `new NODE` là xin cấp phát động một vùng nhớ có kích thước bằng kích thước của kiểu dữ liệu `NODE`.

- Nếu việc cấp phát thành công OS sẽ trả về địa chỉ ô nhớ đầu tiên của vùng nhớ được cấp phát, địa chỉ ô nhớ này được gán cho biến con trỏ `p`.
- Nếu việc cấp phát thất bại, OS sẽ trả về một địa chỉ đặc biệt là địa chỉ NULL, địa chỉ NULL này sẽ được gán cho biến con trỏ `p`.
- Như vậy, thông thường sau câu lệnh thứ 13, biến con trỏ `p` sẽ giữ địa chỉ ô nhớ đầu tiên của vùng nhớ có kích thước bằng kích thước của KDL `NODE`.



Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho danh sách liên kết đơn

```
11. NODE* GetNode(KDL x)
12. {
13.     NODE *p=new NODE;
14.     if(p==NULL)
15.         return NULL;
16.     p->info = x;
17.     p->pNext = NULL;
18.     return p;
19. }
```



TẠO NODE CHO DSLK ĐƠN SỐ NGUYÊN

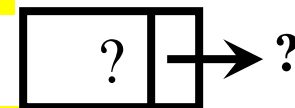


Tạo node cho dslk đơn

- Khái niệm: Tạo node cho danh sách liên kết đơn 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.

Tạo node cho dslk đơn các số nguyên

```
11. NODE* GetNode(int x)
12. {
13.     NODE *p=new NODE;
14.     if(p==NULL)
15.         return NULL;
16.     p->info = x;
17.     p->pNext=NULL;
18.     return p;
19. }
```





Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 04

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



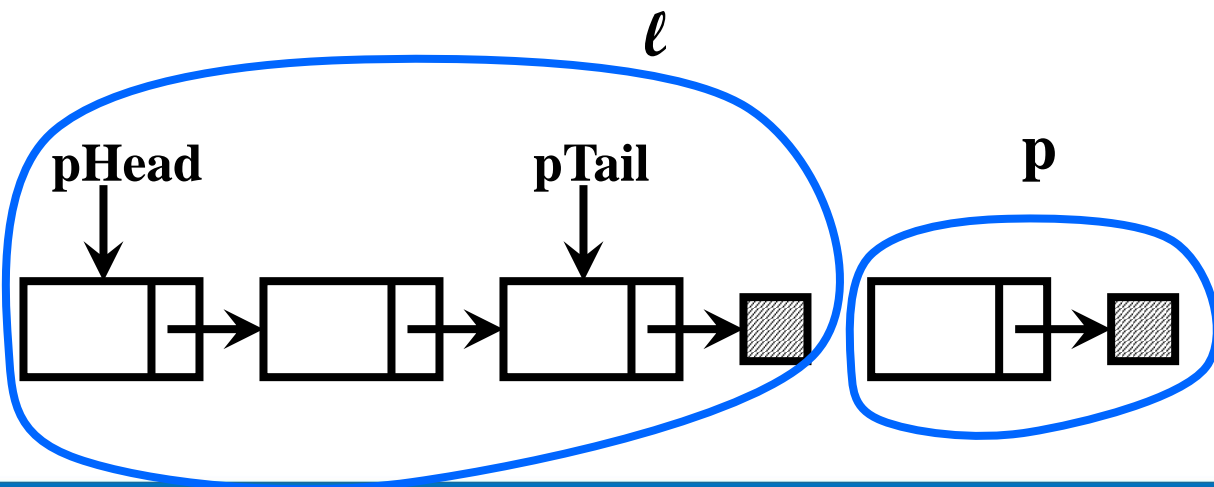
THÊM NODE VÀO CUỐI DSLK ĐƠN



Thêm node vào cuối dslk đơn

— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



— Khai báo hàm

```
11. void AddTail(LIST&, NODE*);
```

— Các câu hỏi:

+ Tên hàm là gì?

+ Tên hàm là AddTail.

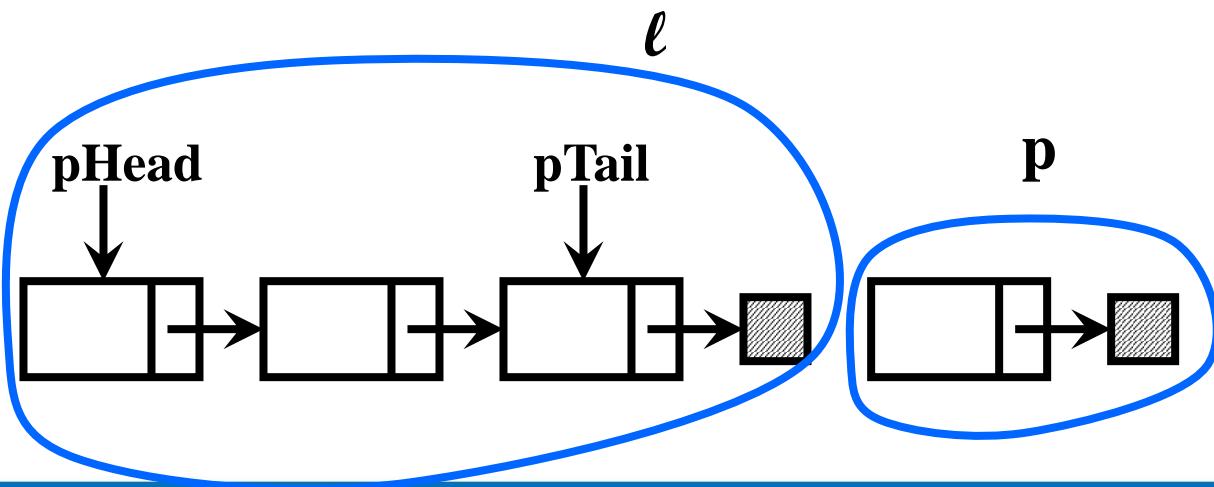
Thêm vào cuối dslk đơn



Thêm node vào cuối dslk đơn

— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



— Khai báo hàm

```
11. void AddTail(LIST&, NODE*);
```

— Các câu hỏi:

+ Kiểu dữ liệu trả về là gì?

+ Kiểu dữ liệu trả về không có.

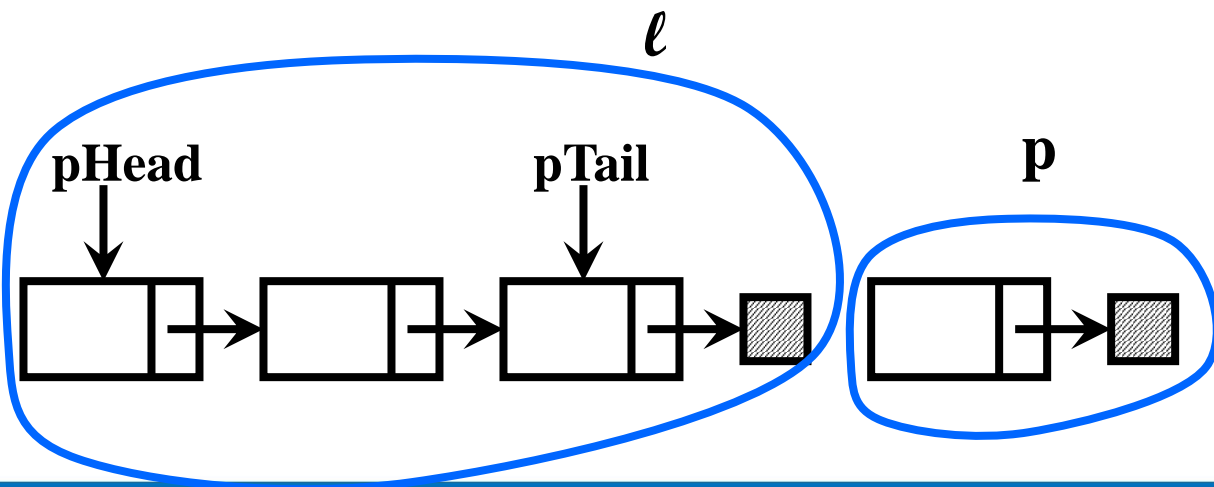
Thêm vào cuối dslk đơn



Thêm node vào cuối dslk đơn

— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



— Khai báo hàm

```
11. void AddTail(LIST&, NODE*);
```

— Các câu hỏi:

+ Có mấy tham số đầu vào?

+ Có hai tham số đầu vào.

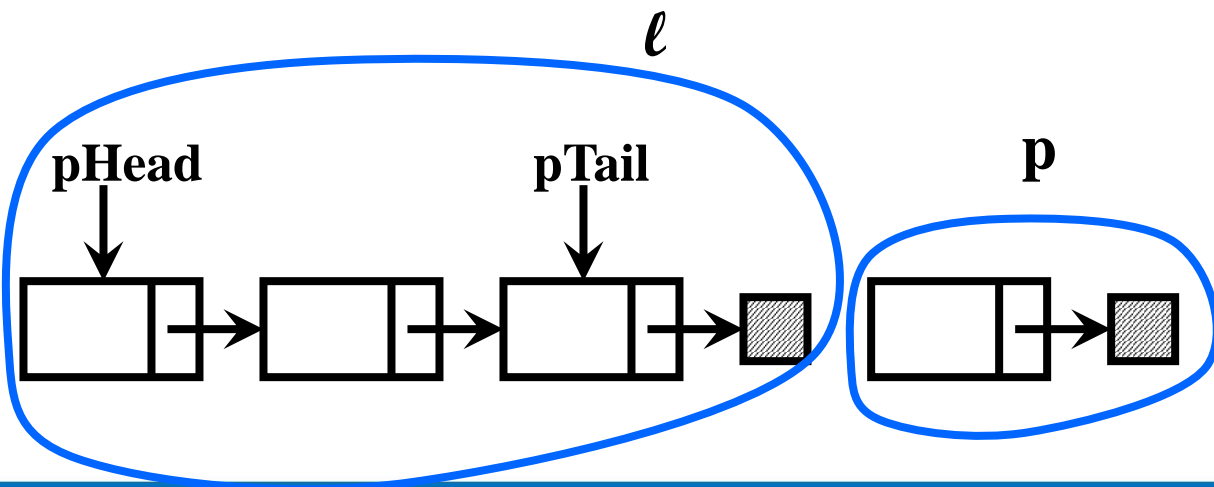
Thêm vào cuối dslk đơn



Thêm node vào cuối dslk đơn

— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



— Khai báo hàm

```
11. void AddTail(LIST&, NODE*);
```

— Các câu hỏi:

+ Kiểu dữ liệu của tham số đầu vào thứ nhất là gì?

+ Kiểu dữ liệu của tham số đầu vào thứ nhất là LIST.

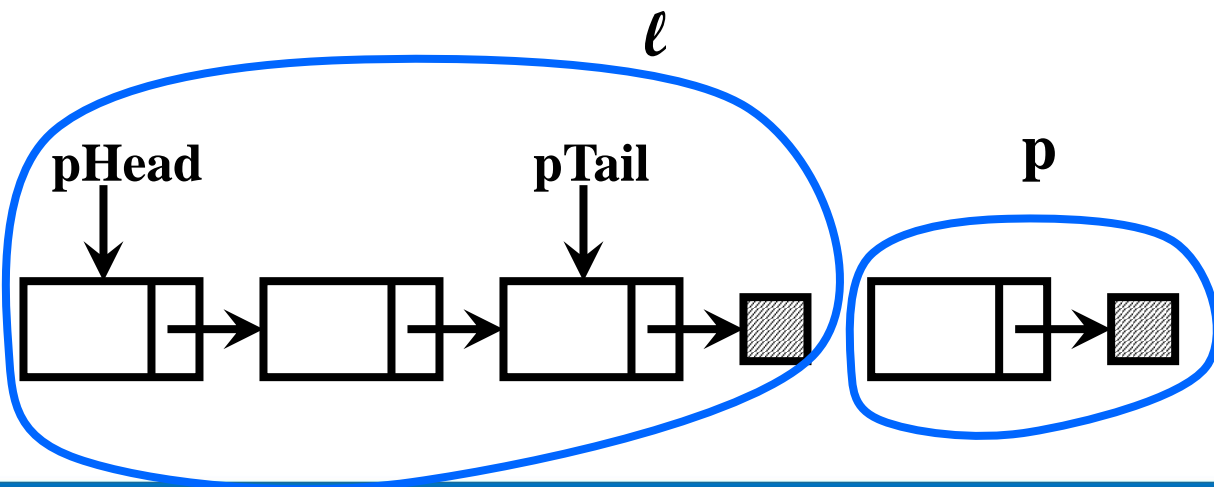
Thêm vào cuối dslk đơn



Thêm node vào cuối dslk đơn

— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



— Khai báo hàm

```
11. void AddTail(LIST&, NODE*);
```

— Các câu hỏi:

+ Loại tham số đầu vào thứ nhất là gì?

+ Loại tham số đầu vào thứ nhất là tham biến.

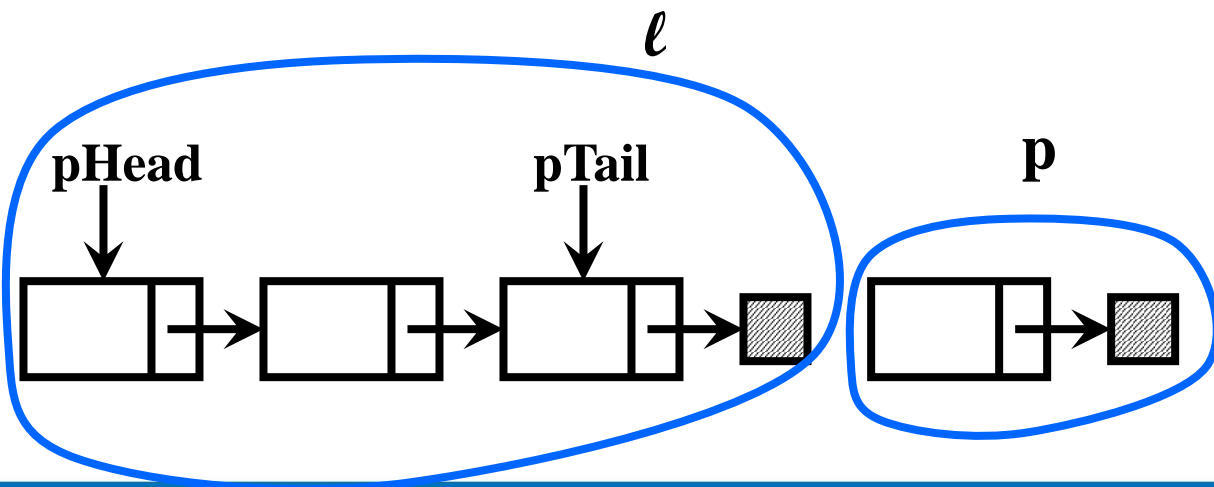
Thêm vào cuối dslk đơn



Thêm node vào cuối dslk đơn

— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



— Khai báo hàm

```
11. void AddTail(LIST&, NODE*);
```

— Các câu hỏi:

+ Kiểu dữ liệu của tham số đầu vào thứ hai là gì?

+ Kiểu dữ liệu của tham số đầu vào thứ hai là con trỏ kiểu cấu trúc NODE.

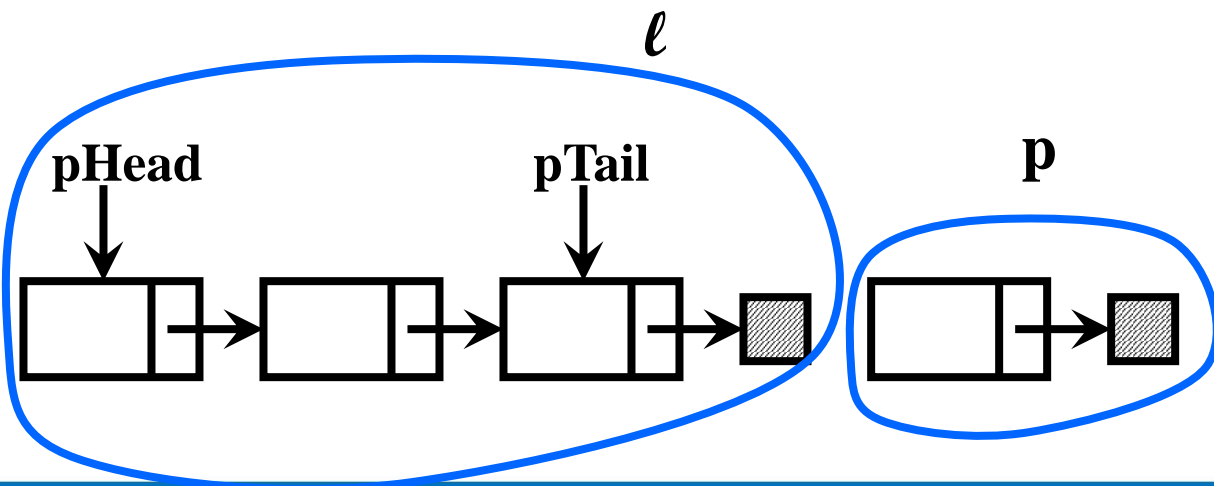
Thêm vào cuối dslk đơn



Thêm node vào cuối dslk đơn

— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



— Khai báo hàm

```
11. void AddTail(LIST&, NODE*);
```

— Các câu hỏi:

+ Loại tham số đầu vào thứ hai là gì?

+ Loại tham số đầu vào thứ hai là tham trị.

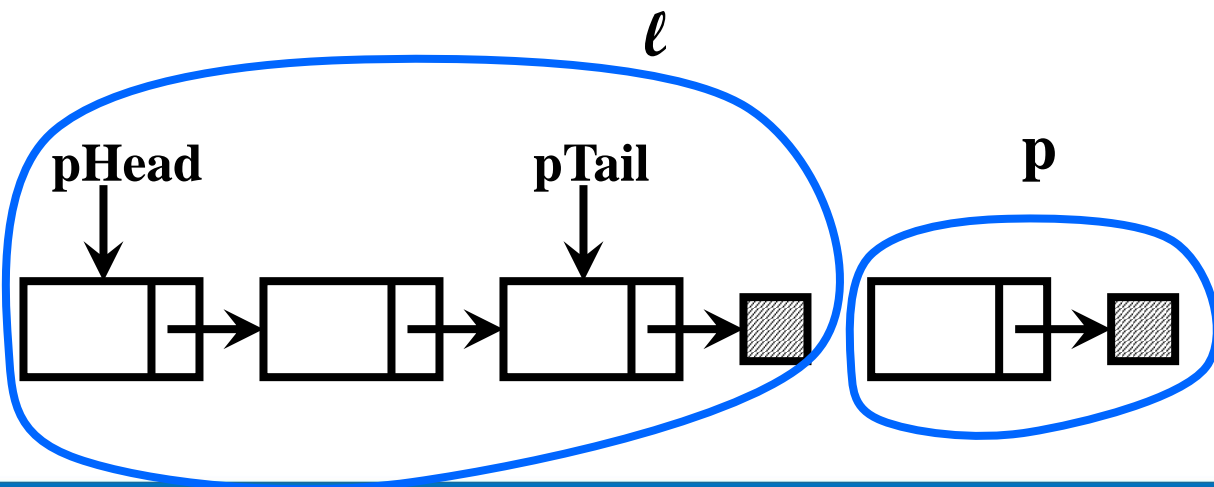
Thêm vào cuối dslk đơn

Thêm node vào cuối dslk đơn



— Khái niệm: Thêm một node vào cuối danh sách liên kết đơn là gắn node đó vào cuối danh sách.

— Hình vẽ



```
11. void AddTail(LIST&l, NODE*p)
12. {
13.     if(l.pHead==NULL)
14.         l.pHead=l.pTail=p;
15.     else
16.     {
17.         l.pTail->pNext=p;
18.         l.pTail = p;
19.     }
20. }
```

Thêm vào cuối dslk đơn



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 05

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



NHẬP TRỮU TỰỢNG DSLK ĐƠN

Nhập trừu tượng dslk đơn



```
11. int Nhap(LIST& l)
12. {
13.     int n;
14.     cout << "Nhap so phan tu: ";
15.     cin >> n;
16.     Init(l);
17.     ...
}
```

Nhập trừu
tượng danh
sách liên kết
đơn

Nhập trờu tợng dslk đơn



```
11. | ...
12. | KDL x;
13. | for (int i = 1; i <= n; i++)
14. | {
15. |     NhapTruuTuong(x);
16. |     NODE* p = GetNode(x);
17. |     if (p != NULL)
18. |         AddTail(l, p);
19. | }
20. | return 1;
21. | }
```

Nhập trờu
tợng danh
sách liên kết
đơn



```
11. int Nhap(LIST& l)
12. {
13.     int n;
14.     cout << "Nhap so phan tu: ";
15.     cin >> n;
16.     Init(l);
17.     KDL x;
18.     for (int i = 1; i <= n; i++)
19.     {
20.         NhapTruuTuong(x);
21.         NODE* p = GetNode(x);
22.         if (p != NULL)
23.             AddTail(l, p);
24.     }
25.     return 1;
26. }
```



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 06

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



NHẬP DSLK ĐƠN CÁC SỐ NGUYÊN TỬ BÀN PHÍM

Nhập dslk đơn số nguyên từ bàn phím



```
11. int Nhap(LIST& l)
12. {
13.     int n;
14.     cout << "Nhap so phan tu: ";
15.     cin >> n;
16.     Init(l);
17.     ...
}
```

Nhập danh sách liên kết đơn các số nguyên từ bàn phím.

Nhập dslk đơn số nguyên từ bàn phím



```
11.  ...
12.  int x;
13.  for (int i = 1; i <= n; i++)
14.  {
15.      cin >> x;
16.      NODE* p = GetNode(x);
17.      if (p != NULL)
18.          AddTail(l, p);
19.  }
20.  return 1;
21. }
```

Nhập danh sách liên kết đơn các số nguyên từ bàn phím.



```
11. int Nhap(LIST& l)
12. {
13.     int n;
14.     cout << "Nhap so phan tu: ";
15.     cin >> n;
16.     Init(l);
17.     int x;
18.     for (int i = 1; i <= n; i++)
19.     {
20.         cout << "Nhap so nguyen thu " << i << ": ";
21.         cin >> x;
22.         NODE* p = GetNode(x);
23.         if (p != NULL)
24.             AddTail(l, p);
25.     }
26.     return l;
27. }
```



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 07

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



TẠO NGẪU NHIÊN DSLK ĐƠN CÁC SỐ NGUYÊN

Tạo ngẫu nhiên dslk đơn số nguyên



```
11. int TaoNgauNhien(LIST& l)
12. {
13.     int n;
14.     cout << "Nhap so phan tu: ";
15.     cin >> n;
16.     Init(l);
17.     srand(time(NULL));
18.     ...
```

Tạo ngẫu
nhiên danh
sách liên kết
đơn các số
nguyên.

Tạo ngẫu nhiên dslk đơn số nguyên



```
11. | ...
12. | int x;
13. | for (int i = 1; i <= n; i++)
14. | {
15. |     x = rand()%(200+1) - 100;
16. |     NODE* p = GetNode(x);
17. |     if (p != NULL)
18. |         AddTail(l, p);
19. | }
20. | return l;
21. | }
```

Tạo ngẫu
nhiên danh
sách liên kết
đơn các số
nguyên.



```
11. int TaoNgauNhien(LIST& l)
12. {
13.     int n;
14.     cout << "Nhap so phan tu: ";
15.     cin >> n;
16.     Init(l);
17.     srand(time(NULL));
18.     int x;
19.     for (int i = 1; i <= n; i++)
20.     {
21.         x = rand()%(200+1) - 100;
22.         NODE* p = GetNode(x);
23.         if (p != NULL)
24.             AddTail(l, p);
25.     }
26.     return 1;
27. }
```



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 08

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



NHẬP DSLK ĐƠN CÁC SỐ NGUYÊN TỬ FILE

Nhập dslk đơn các số nguyên từ file



— Định nghĩa hàm nhập dslk đơn từ tập tin có tên intdata01.inp

A screenshot of a Notepad window titled '*intdata01.inp - Notepad'. The window has a menu bar with 'File', 'Edit', 'Format', 'View', and 'Help'. The text content of the file is as follows:

```
10
24 56 53 44 -54 6 63 -47 91 -99
```

Nhập dslk đơn các số nguyên từ file



— Định dạng tập tin `intdata01.inp`

+ Dòng đầu tiên: số phần tử của dslk đơn (n).

+ Dòng tiếp theo: lưu n số nguyên tương ứng với các giá trị trong dslk đơn.



*intdata01.inp - Notepad

File Edit Format View Help

10

24 56 53 44 -54 6 63 -47 91 -99

Nhập dslk đơn các số nguyên từ file



```
11. int Nhap(LIST&l, string filename)
```

```
12. {
```

```
13.     ifstream fi(filename);
```

Nhập danh
sách liên
kết đơn từ
file

Nhập dslk đơn các số nguyên từ file



```
11. int Nhap(LIST&l, string filename)
```

```
12. {
```

```
13.     ifstream fi(filename);
```

Ý1: *fi* là đối tượng thuộc lớp
ifstream.

Nhập danh
sách liên
kết đơn từ
file

Nhập dslk đơn các số nguyên từ file



```
11. int Nhap(LIST&l, string filename)
```

```
12. {
```

```
13.     ifstream fi(filename);
```

Ý2: Dòng lệnh số 13 khai báo đối tượng **fi** với đối số có tên **filename** và có kiểu **string**.

Nhập danh
sách liên
kết đơn từ
file

Nhập dslk đơn các số nguyên từ file



```
11. int Nhap(LIST&l, string filename)
```

```
12. {
```

```
13.     ifstream fi(filename);
```

Ý3: Khi chương trình thực hiện tới dòng lệnh số 13. Đối tượng **fi** gọi thực hiện phương thức thiết lập với tham số có kiểu **string**. Phương thức thiết lập sẽ mở tập tin có tên lưu trong biến **filename**.

Nhập danh
sách liên
kết đơn từ
file

Nhập dslk đơn các số nguyên từ file



```
11. int Nhap(LIST&l, string filename)
12. {
13.     ifstream fi(filename);
14.     if (fi.fail() == true)
15.         return 0;
16.     int n;
17.     int x;
18.     fi >> n;
19.     Init(l);
20.     ...
```

Nhập danh
sách liên
kết đơn từ
file

Nhập dslk đơn các số nguyên từ file



```
11. | ...
12. | for (int i = 1; i <= n; i++)
13. | {
14. |     fi >> x;
15. |     NODE* p = GetNode(x);
16. |     if (p != NULL)
17. |         AddTail(l, p);
18. | }
19. | return 1;
20. | }
```

Nhập danh
sách liên
kết đơn từ
file



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 09

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang

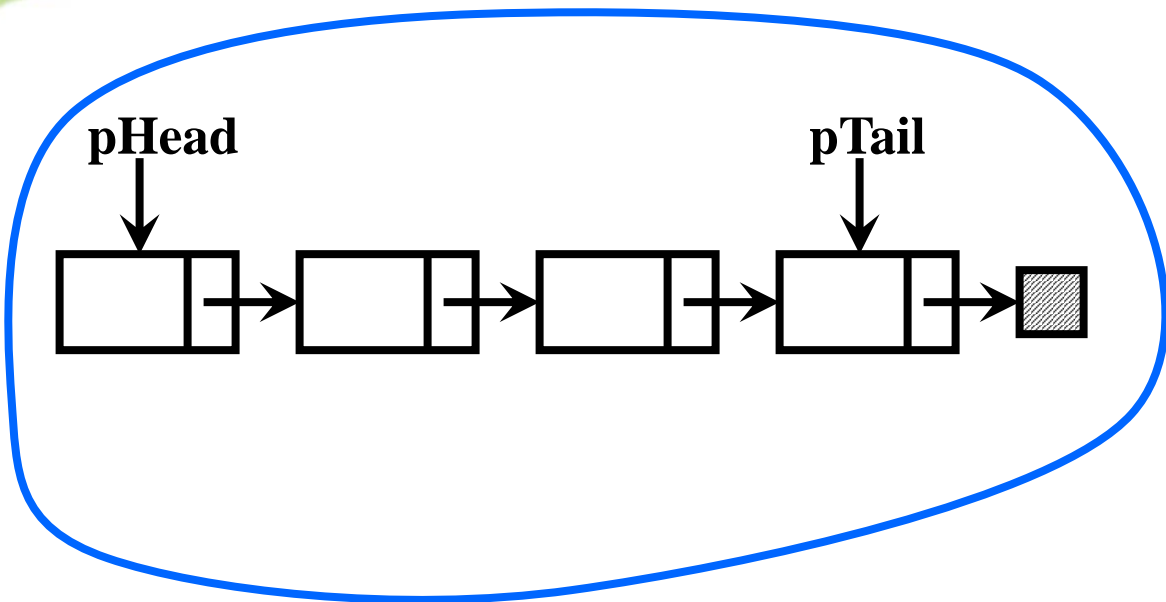


XUẤT DSLK ĐƠN CÁC SỐ NGUYÊN

Xuất dslk đơn các số nguyên



LIST



Xuất danh sách liên
kết đơn

```
11. void Xuat(LIST l)
12. {
13.     NODE* p = l.pHead;
14.     while (p != NULL)
15.     {
16.         cout<<p->info;
17.         p = p->pNext;
18.     }
19. }
```



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 10

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang

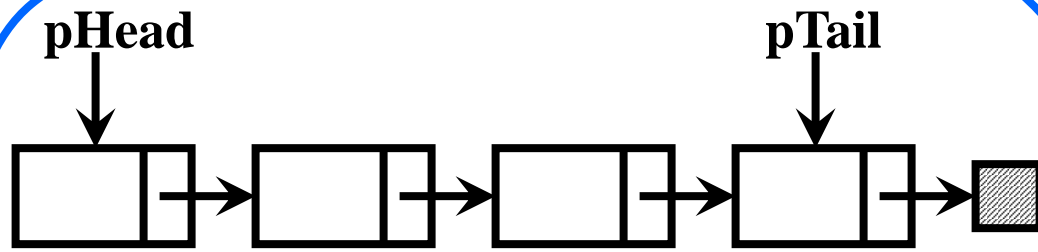


ĐẾM NODE TRONG DSLK ĐƠN

Đếm node trong dslk đơn



LIST



Đếm node trong danh sách liên kết đơn

```
11. int DemNode(LIST l)
12. {
13.     int dem = 0;
14.     NODE* p = l.pHead;
15.     while (p != NULL)
16.     {
17.         dem++;
18.         p = p->pNext;
19.     }
20.     return dem;
21. }
```




Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 11

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang

Xuất dslk đơn các số nguyên ra file



— Định nghĩa hàm xuất dslk đơn ra tập tin có tên intdata01.out

A screenshot of a Notepad window titled '*intdata01.inp - Notepad'. The window has a menu bar with 'File', 'Edit', 'Format', 'View', and 'Help'. The text content consists of two lines: '10' on the first line and '24 56 53 44 -54 6 63 -47 91 -99' on the second line.

```
*intdata01.inp - Notepad
File Edit Format View Help
10
24 56 53 44 -54 6 63 -47 91 -99
```

Xuất dslk đơn các số nguyên ra file



— Định dạng tập tin `intdata01.out`

+ Dòng đầu tiên: số phần tử của dslk đơn (n).

+ Dòng tiếp theo: lưu n số nguyên tương ứng với các giá trị trong dslk đơn.



*intdata01.inp - Notepad

File Edit Format View Help

10

24 56 53 44 -54 6 63 -47 91 -99



XUẤT DSLK ĐƠN CÁC SỐ NGUYÊN RA FILE

```
11.int Xuat(LIST l, string filename)
12.{
13.    ofstream fo(filename);
24.}
```



Xuất danh
sách liên
kết đơn ra
file

```
11.int Xuat(LIST l, string filename)
12.{
13.    ofstream fo(filename);

    Ý1: fo là đối tượng thuộc lớp
    ofstream.

24.}
```



Xuất danh
sách liên
kết đơn ra
file

```
11.int Xuat(LIST l, string filename)
12.{
13.    ofstream fo(filename);

    Ý2: Dòng lệnh số 13 khai báo
    đối tượng fo với đối số có tên
    filename và có kiểu string.

24.}
```



Xuất danh
sách liên
kết đơn ra
file


```
11.int Xuat(LIST l, string filename)
12.{
13.    ofstream fo(filename);

    Ý3: Khi chương trình thực hiện
    tới dòng lệnh số 13. Đối tượng
    fo gọi thực hiện phương thức
    thiết lập với tham số có kiểu
    string. Phương thức thiết lập
    sẽ mở tập tin có tên lưu trong
    biến filename.

24.}
```



Xuất danh
sách liên
kết đơn ra
file



```
11. int Xuat(LIST l, string filename)
12. {
13.     ofstream fo(filename);
14.     if (fo.fail()==true)
15.         return 0;
16.     fo << setw(5) << DemNode(1) << endl;
17.     NODE* p = l.pHead;
18.     while (p != NULL)
19.     {
20.         fo << setw(5) << p->info;
21.         p = p->pNext;
22.     }
23.     return 1;
24. }
```

Xuất danh
sách liên
kết đơn ra
file



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 12

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang

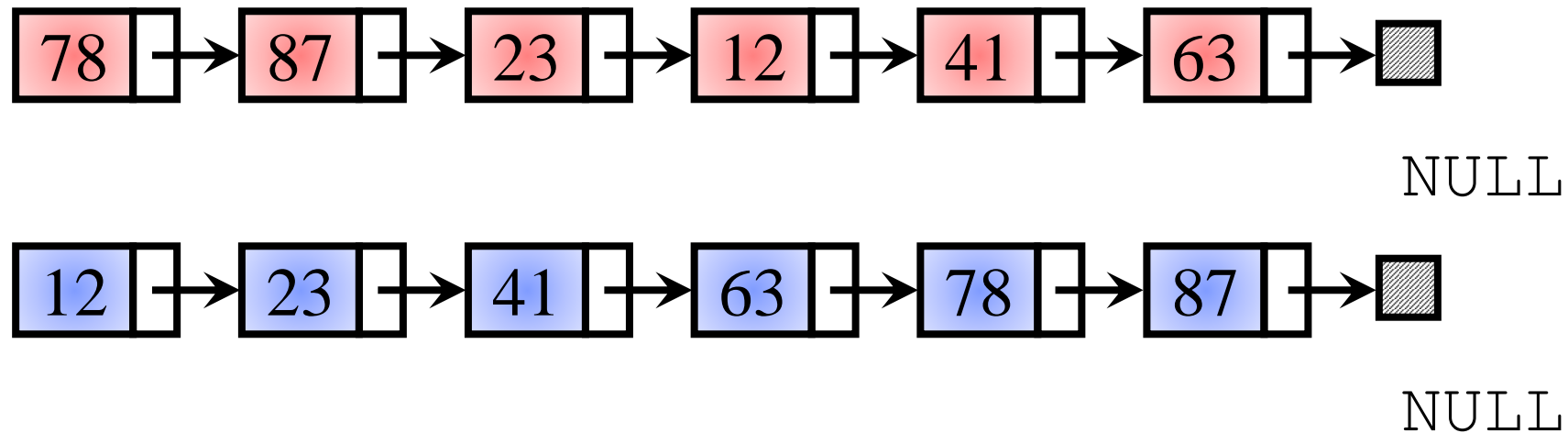


SẮP TĂNG DSLK ĐƠN CÁC SỐ NGUYÊN

Sắp tăng dslk đơn các số nguyên



- Bài toán: Định nghĩa hàm sắp xếp danh sách liên kết đơn các số nguyên tăng dần bằng thuật toán **Interchange sort**.



Sắp tăng dslk đơn các số nguyên



—Thuật toán interchange sort sẽ duyệt qua tất cả các cặp giá trị trong không gian cần sắp xếp và hoán vị hai giá trị trong một cặp nếu cặp giá trị đó là nghịch thế.

Sắp tăng dslk đơn các số nguyên



```
11. void InterchangeSort(LIST& l)
12. {
13.     for (NODE*p=l.pHead; p->pNext!=NULL; p=p->pNext)
14.         for (NODE*q=p->pNext; q!=NULL; q=q->pNext)
15.             if (p->info > q->info)
16.                 swap(p->info, q->info);
17. }
```

Sắp xếp danh sách
liên kết đơn tăng dần



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 13

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



DỰ ÁN 01 DANH SÁCH LIÊN KẾT ĐƠN

Dự án danh sách liên kết đơn



- Viết chương trình thực hiện các yêu cầu sau:
 - + Nhập dslk đơn các số nguyên từ bàn phím;
 - + Sắp xếp dslk đơn các số nguyên tăng dần bằng thuật toán interchange sort.
 - + Xuất dslk đơn các số nguyên sau khi sắp xếp ra màn hình.

Kiến trúc chương trình



```
11.#include <iostream>
12.#include <fstream>
13.#include <ctime>
14.#include <iomanip>
15.#include <string>
16.using namespace std;
```

Khai báo
sử dụng
thư viện

Kiến trúc chương trình



```
11.struct node
12.{
13.|    int info;
14.|    struct node* pNext;
15.};
16.typedef struct node NODE;
17.struct list
18.{
19.|    NODE* pHead;
20.|    NODE* pTail;
21.};
22.typedef struct list LIST;
```

Khai báo
cấu trúc
dữ liệu

Kiến trúc chương trình



```
11.void Init(LIST& );  
12.int IsEmpty(LIST );  
13.NODE* GetNode(int );  
14.void AddTail(LIST& , NODE* );  
15.int Nhap(LIST&);  
16.void Xuat(LIST );  
  
17.void InterchangeSort(LIST& );
```

Khai báo
hàm



```
11. int main()  
12. {  
13.     LIST lst;  
14.     Nhap(lst);  
15.     cout << "\n Danh sach ban dau: ";  
16.     Xuat(lst);  
17.     InterchangeSort(lst);  
18.     cout << "\n Danh sach sau khi sap tang: ";  
19.     Xuat(lst);  
20.     return 1;  
21. }
```

Định nghĩa hàm
main



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 14

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



DỰ ÁN 02 DANH SÁCH LIÊN KẾT ĐƠN

Dự án danh sách liên kết đơn



- Viết chương trình thực hiện các yêu cầu sau:
 - + Tạo ngẫu nhiên dslk đơn các số nguyên.
 - + Sắp xếp dslk đơn các số nguyên tăng dần bằng thuật toán interchange sort.
 - + Xuất dslk đơn các số nguyên sau khi sắp xếp ra màn hình.

Kiến trúc chương trình



```
11.#include <iostream>
12.#include <fstream>
13.#include <ctime>
14.#include <iomanip>
15.#include <string>
16.using namespace std;
```

Khai báo
sử dụng
thư viện

Kiến trúc chương trình



```
11.struct node
12.{
13.|    int info;
14.|    struct node* pNext;
15.};
16.typedef struct node NODE;
17.struct list
18.{
19.|    NODE* pHead;
20.|    NODE* pTail;
21.};
22.typedef struct list LIST;
```

Khai báo
cấu trúc
dữ liệu

Kiến trúc chương trình



```
11.void Init(LIST& );  
12.int IsEmpty(LIST );  
13.NODE* GetNode(int );  
14.void AddTail(LIST& , NODE* );  
15.int TaoNgauNhiem(LIST&);  
16.void Xuat(LIST );  
  
17.void InterchangeSort(LIST& );
```

Khai báo
hàm



```
11. int main()  
12. {  
13.     LIST lst;  
14.     TaoNgauNhien(lst);  
15.     cout << "\n Danh sach ban dau: ";  
16.     Xuat(lst);  
17.     InterchangeSort(lst);  
18.     cout << "\n Danh sach sau khi sap tang: ";  
19.     Xuat(lst);  
20.     return 1;  
21. }
```

Định nghĩa hàm
main



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 15

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



DỰ ÁN 03 DANH SÁCH LIÊN KẾT ĐƠN

Dự án danh sách liên kết đơn



- Viết chương trình thực hiện các yêu cầu sau:
 - + Nhập dslk đơn các số nguyên từ tập tin: `intdata01.inp`.
 - + Sắp xếp dslk đơn các số nguyên tăng dần bằng thuật toán interchange sort.
 - + Xuất dslk đơn các số nguyên sau khi sắp xếp ra các tập tin: `intdata01.out`.

Dự án danh sách liên kết đơn



- Định dạng tập tin `intdata01.inp` và `intdata01.out`
 - + Dòng đầu tiên: số phần tử của dslk đơn các số nguyên (n).
 - + Dòng tiếp theo: lưu n số nguyên tương ứng với các giá trị trong dslk đơn các số nguyên.

Kiến trúc chương trình



```
11.#include <iostream>
12.#include <fstream>
13.#include <iomanip>
14.#include <string>
15.using namespace std;
```

Khai báo
sử dụng
thư viện

Kiến trúc chương trình



```
11.struct node
12.{
13.|    int info;
14.|    struct node* pNext;
15.};
16.typedef struct node NODE;
17.struct list
18.{
19.|    NODE* pHead;
20.|    NODE* pTail;
21.};
22.typedef struct list LIST;
```

Khai báo
cấu trúc
dữ liệu

Kiến trúc chương trình



```
11.void Init(LIST& );  
12.int IsEmpty(LIST );  
13.NODE* GetNode(int );  
14.void AddTail(LIST& , NODE* );  
15.int Nhap(LIST& , string );  
16.void InterchangeSort(LIST& );  
17.void Xuat(LIST );  
18.int DemNode(LIST );  
19.int Xuat(LIST , string );
```

Khai báo
hàm



```
11.int main()  
12.{  
13.    LIST lst;  
14.    if( Nhap(lst, "intdata01.inp") == 1)  
15.    {  
16.        InterchangeSort(lst);  
17.        Xuat(lst, "intdata01.out");  
18.    }  
19.    else  
20.        cout << "Loi mo file";  
21.    return 1;  
22.}
```

Định nghĩa hàm
main



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang



DANH SÁCH LIÊN KẾT ĐƠN – PHẦN 16

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



DỰ ÁN 04 DANH SÁCH LIÊN KẾT ĐƠN

Dự án danh sách liên kết đơn



- Viết chương trình thực hiện các yêu cầu sau:
 - + Nhập dslk đơn các số nguyên từ các tập tin: intdata01.inp; intdata02.inp; ...; intdata09.inp; intdata10.inp; intdata11.inp; intdata12.inp; intdata13.inp;
 - + Sắp xếp dslk đơn các số nguyên tăng dần bằng thuật toán interchange sort.
 - + Xuất dslk đơn các số nguyên sau khi sắp xếp ra các tập tin: intdata01.out; intdata02.out; ...; intdata09.out; intdata10.out; intdata11.out; intdata12.out; intdata13.out;

Dự án danh sách liên kết đơn



- Định dạng tập tin `intdataxx.inp` và `intdataxx.out`
 - + Dòng đầu tiên: số phần tử của dslk đơn các số nguyên (n).
 - + Dòng tiếp theo: lưu n số nguyên tương ứng với các giá trị trong dslk đơn các số nguyên.

Kiến trúc chương trình



```
11.#include <iostream>
12.#include <fstream>
13.#include <iomanip>
14.#include <string>
15.using namespace std;
```

Khai báo
sử dụng
thư viện

Kiến trúc chương trình



```
11.struct node
12.{
13.|    int info;
14.|    struct node* pNext;
15.};
16.typedef struct node NODE;
17.struct list
18.{
19.|    NODE* pHead;
20.|    NODE* pTail;
21.};
22.typedef struct list LIST;
```

Khai báo
cấu trúc
dữ liệu

Kiến trúc chương trình



```
11.void Init(LIST& );  
12.int IsEmpty(LIST );  
13.NODE* GetNode(int );  
14.void AddTail(LIST& , NODE* );  
15.int Nhap(LIST& , string );  
16.void InterchangeSort(LIST& );  
17.void Xuat(LIST );  
18.int DemNode(LIST );  
19.int Xuat(LIST , string );
```

Khai báo
hàm



```
11. int main()
12. {
13.     LIST lst;
14.     for (int i = 1; i <= 13; i++)
15.     {
16.         string filename = "intdata";
17.         if (i < 10)
18.             filename += '0';
19.         filename += to_string(i);
20.         string filenameinp = filename;
21.         filenameinp += ".inp";
22.         if (Nhap(lst, filenameinp) == 1)
23.         {
24.             InterchangeSort(lst);
25.             string filenameout = filename;
26.             filenameout += ".out";
27.             Xuat(lst, filenameout);
28.             cout << "\n" << filenameinp;
29.             cout << "\n" << filenameout;
30.         }
31.         else
32.             cout << "\n Khong mo duoc file " << filename << "\n";
33.     }
34.     cout << "\n\n\n";
35.     return 1;
36. }
```

Định nghĩa hàm main



Cảm ơn quý vị đã lắng nghe

Nhóm tác giả

Hồ Thái Ngọc

ThS. Võ Duy Nguyên

TS. Nguyễn Tấn Trần Minh Khang