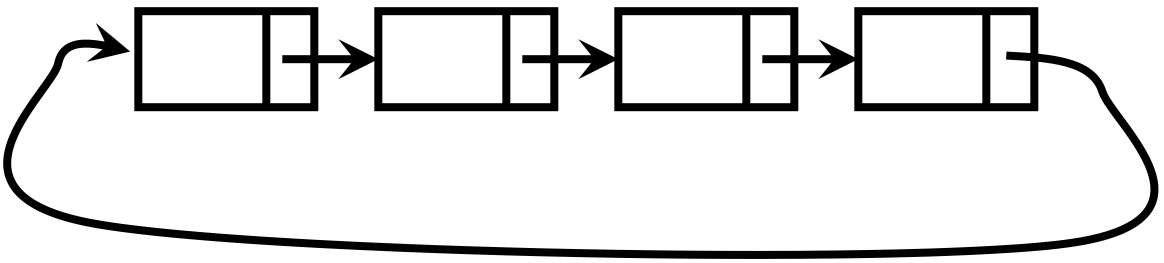
Chương 5
CẤU TRÚC DỮ LIỆU DANH SÁCH LIÊN KẾT ĐƠN VÒNG

TS. Nguyễn Tấn Trần Minh Khang

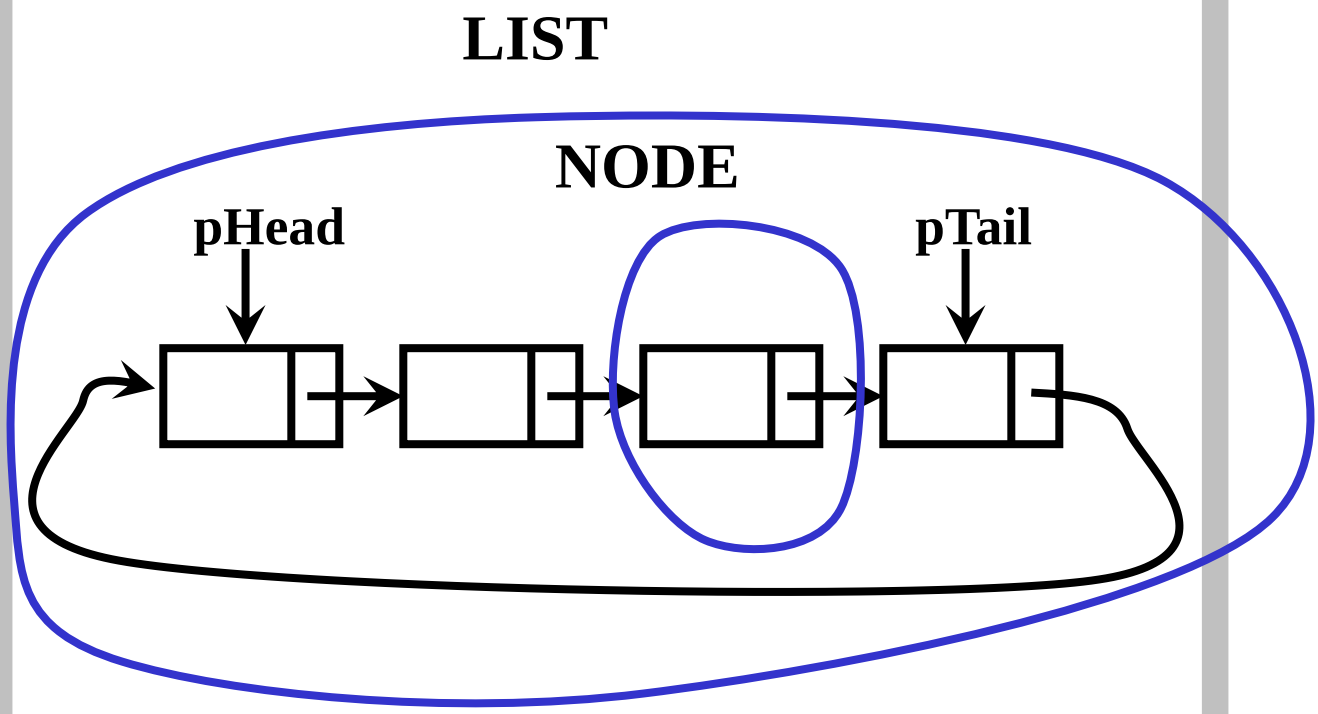
ThS. Cáp Phạm Đình Thăng

1. HÌNH ẢNH DSLK ĐƠN VÒNG



2. CẤU TRÚC DỮ LIỆU DSLK ĐƠN VÒNG

- Hình vẽ



- Khai báo cấu trúc dữ liệu

2. CẤU TRÚC DỮ LIỆU DSLK ĐƠN VÒNG

```
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;
```

–KDL là kiểu dữ liệu của đối tượng được lưu trong danh sách liên kết đơn.

2. CẤU TRÚC DỮ LIỆU DSLK ĐƠN VÒNG

– **Ví dụ 1: Hãy khai báo CTDL cho dslk đơn vòng 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;
```

2. CẤU TRÚC DỮ LIỆU DSLK ĐƠN VÒNG

– **Ví dụ 2: Hãy khai báo CTDL cho dslk đơn vòng 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;
```

2. CẤU TRÚC DỮ LIỆU DSLK ĐƠN VÒNG

– **Ví dụ 3: Hãy khai báo ctdl cho dslk đơn vòng các phân số.**

```
11.struct phanso
12.{
13.|    int tu;
14.|    int mau;
15.};
16.typedef struct phanso PHANSO;
17.struct node
18.{
19.|    PHANSO info;
20.|    struct node*pNext;
21.};
22.typedef struct node NODE;
23.struct list
24.{
25.|    NODE*pHead;
26.|    NODE*pTail;
27.};
28.typedef struct list LIST;
```

2. CẤU TRÚC DỮ LIỆU DSLK ĐƠN VÒNG

– **Ví dụ 4: Hãy khai báo ctdl cho dslk đơn vòng tọa độ các điểm trong mp Oxy.**

```
11.struct diem
12.{
13.|    float x;
14.|    float y;
15.};
16.typedef struct diem DIEM;
17.struct node
18.{
19.|    DIEM info;
20.|    struct node*pNext;
21.};
22.typedef struct node NODE;
23.struct list
24.{
25.|    NODE*pHead;
26.|    NODE*pTail;
27.};
28.typedef struct list LIST;
```

3. KHỞI TẠO DSLK ĐƠN VÒNG

- **Khái niệm: Khởi tạo danh sách liên kết đơn vòng là tạo ra danh sách rỗng không chứa node nào hết.**
- Định nghĩa hàm

```
1. void Init(LIST &ℓ)
2. {
3.     ℓ.pHead = NULL;
4.     ℓ.pTail = NULL;
5. }
```

4. KIỂM TRA DSLK ĐƠN VÒNG RỖNG

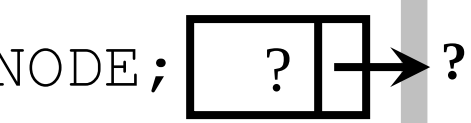
- **Khái niệm: Kiểm tra danh sách liên kết đơn vòng rỗng là hàm trả về giá trị 1 khi danh sách rỗng. Trong tình huống danh sách không rỗng thì hàm sẽ trả về giá trị 0.**
- Định nghĩa hàm

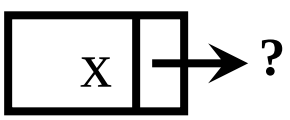
```
1. int IsEmpty(LIST ℓ)
2. {
3.     if (ℓ.pHead==NULL)
4.         return 1;
5.     return 0;
6. }
```

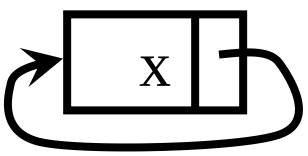
5. TẠO NODE CHO DSLK ĐƠN VÒNG

- **Khái niệm: Tạo node cho danh sách liên kết đơn vòng 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.**
- Định nghĩa hàm trừu tượng

```
1. NODE* GetNode(KDL x)
2. {
3.     NODE *p = new NODE;
4.     if (p==NULL)
5.         return NULL;
6.     p->info = x;
7.     p->pNext = p;
8.     return p;
9. }
```







5. TẠO NODE CHO DSLK ĐƠN VÒNG

- **Ví dụ 1: Định nghĩa hàm tạo một NODE cho dslk đơn vòng các số nguyên để chứa thông tin đã được biết trước.**
- Định nghĩa hàm

```
1. NODE* GetNode(int x)
2. {
3.     NODE *p = new NODE;
4.     if (p==NULL)
5.         return NULL;
6.     p->info = x;
7.     p->pNext = p;
8.     return p;
9. }
```

5. TẠO NODE CHO DSLK ĐƠN VÒNG

- **Ví dụ 2: Định nghĩa hàm tạo một NODE cho dslk đơn vòng các số thực để chứa thông tin đã được biết trước.**
- Định nghĩa hàm

```
1. NODE* GetNode(float x)
2. {
3.     NODE *p = new NODE;
4.     if (p==NULL)
5.         return NULL;
6.     p->info = x;
7.     p->pNext = p;
8.     return p;
9. }
```

5. TẠO NODE CHO DSLK ĐƠN VÒNG

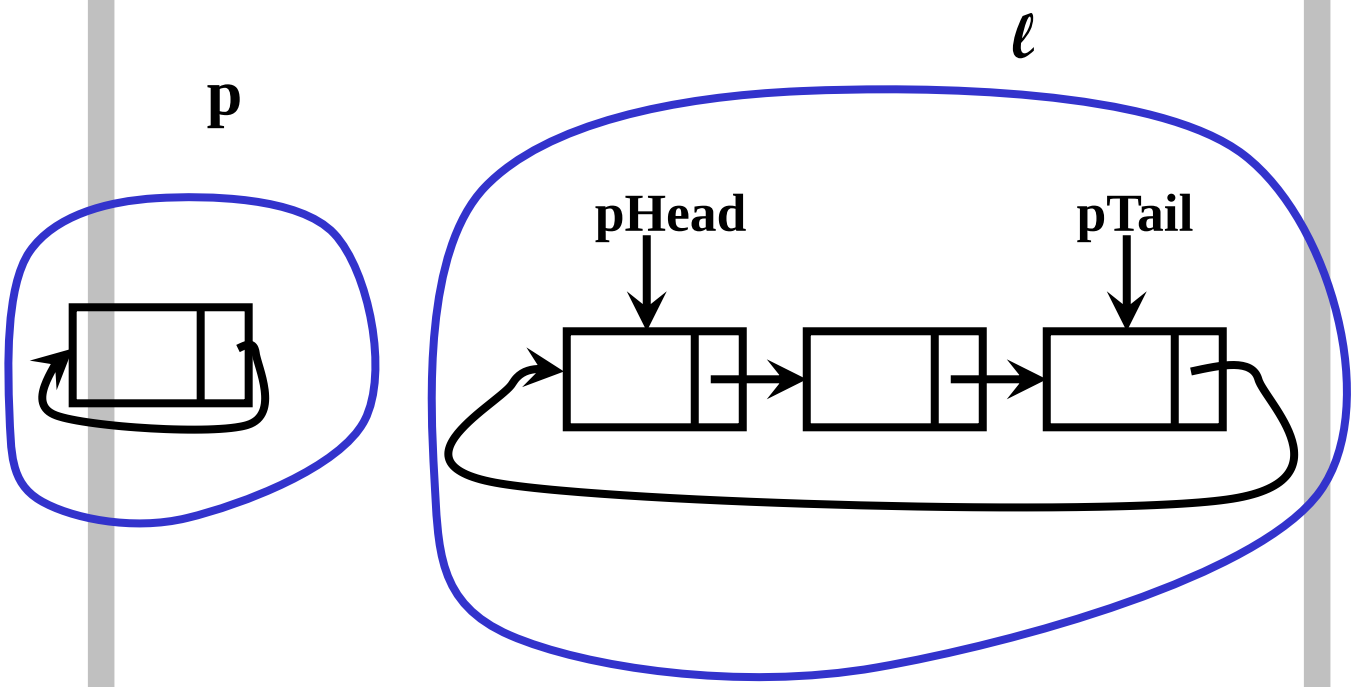
- **Ví dụ 4: Định nghĩa hàm tạo một NODE cho dslk đơn vòng tọa độ các điểm để chứa thông tin đã được biết trước.**

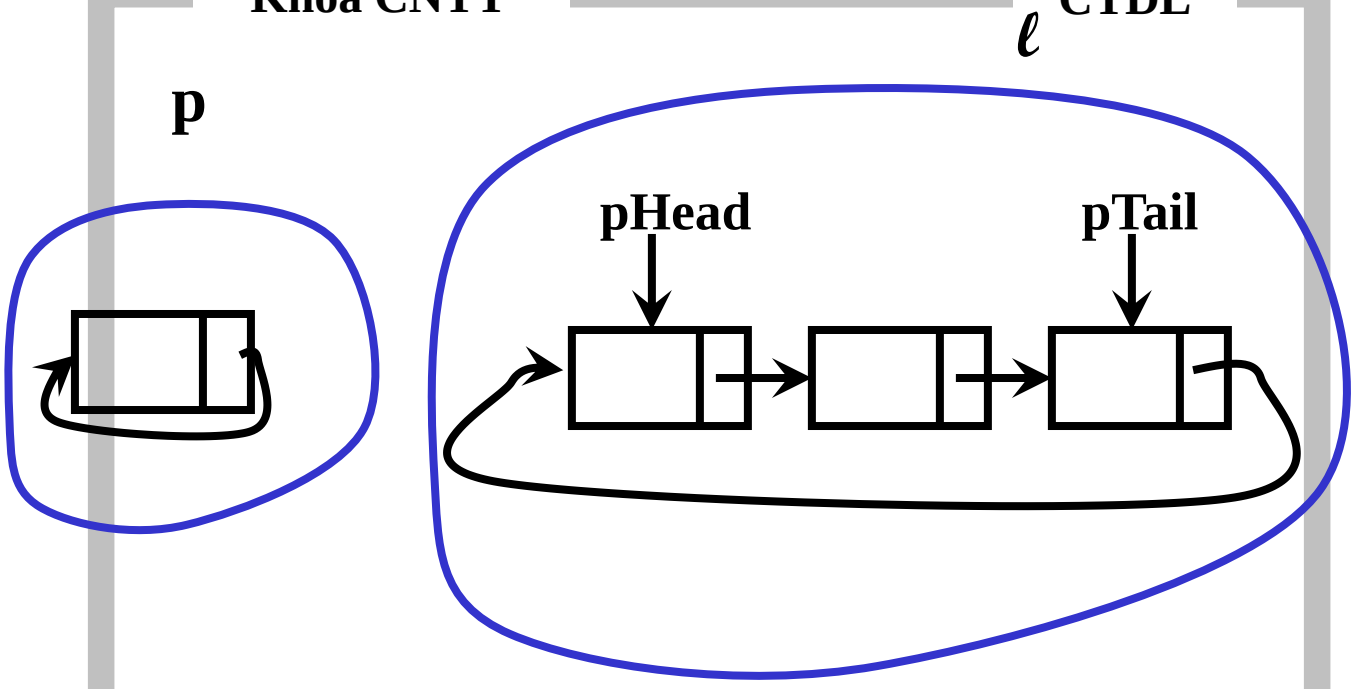
– Định nghĩa hàm

```
1. NODE* GetNode(DIEM P)
2. {
3.     NODE *p = new NODE;
4.     if (p==NULL)
5.         return NULL;
6.     p->info = P;
7.     p->pNext = p;
8.     return p;
9. }
```

6. THÊM MỘT NODE VÀO ĐẦU DSLK ĐƠN VÒNG

- **Khái niệm: Thêm một node vào đầu danh sách liên kết đơn vòng là gắn node đó vào đầu danh sách.**
- Hình vẽ





- Định nghĩa hàm:

```
11. void AddHead(LIST&ℓ,NODE*p)
12. {
13.     if(ℓ.pHead==NULL)
14.         ℓ.pHead = ℓ.pTail = p;
15.     else
16.     {
17.         p->pNext = ℓ.pHead;
18.         ℓ.pTail->pNext = p;
19.         ℓ.pHead = p;
20.     }
21. }
```

7. NHẬP TỪ BÀN PHÍM DSLK ĐƠN VÒNG

- **Khái niệm: Nhập từ bàn phím dslk đơn vòng là lần lượt nhập các thông tin của từng node trong dslk.**
- Định nghĩa hàm trừu tượng

```
11. void Input(LIST&ℓ)
12. {
13.     int n;
14.     printf("Nhap n: ");
15.     scanf("%d",&n);
16.     Init(ℓ);
17.     for(int i=1;i<=n;i++)
18.     {
19.         KDL x;
20.         Nhap(x)
21.         NODE*p = GetNode(x);
22.         if (p!=NULL)
23.             AddHead(ℓ,p);
24.     }
25. }
```

7. NHẬP TỪ BÀN PHÍM DSLK ĐƠN VÒNG

- **Ví dụ 1: Nhập danh sách liên kết đơn vòng các số nguyên.**
- Định nghĩa hàm

```
11. void Input(LIST&ℓ)
12. {
13.     int n;
14.     printf("Nhap n: ");
15.     scanf("%d",&n);
16.     Init(ℓ);
17.     for(int i=1;i<=n;i++)
18.     {
19.         int x;
20.         printf("Nhap so: ");
21.         scanf("%d",&x);
22.         NODE*p = GetNode(x);
23.         if (p!=NULL)
24.             AddHead(ℓ,p);
25.     }
26. }
```

7. NHẬP TỪ BÀN PHÍM DSLK ĐƠN VÒNG

- **Ví dụ 2: Nhập danh sách liên kết đơn vòng các số thực.**
- Định nghĩa hàm

```
11. void Input(LIST&ℓ)
12. {
13.     int n;
14.     printf("Nhap n: ");
15.     scanf("%d",&n);
16.     Init(ℓ);
17.     for(int i=1;i<=n;i++)
18.     {
19.         float x;
20.         printf("Nhap so: ");
21.         scanf("%f",&x);
22.         NODE*p = GetNode(x);
23.         if (p!=NULL)
24.             AddHead(ℓ,p);
25.     }
26. }
```

8. DUYỆT TUẦN TỰ DSLK ĐƠN VÒNG

- **Khái niệm: duyệt danh sách liên kết đơn vòng là thăm qua tất cả các node mỗi node một lần.**
- Định nghĩa hàm trừu tượng

```
KDL <Tên Hàm>(LIST ℓ)
{
    if (ℓ.pHead==NULL)
        return ...
    ...
    NODE*p = ℓ.pHead;
    do{
        ...
        p = p->pNext;
    } while(p!=ℓ.pHead);
    ...
}
```

8. DUYỆT TUẦN TỰ DSLK ĐƠN VÒNG

- **Ví dụ 1: Định nghĩa hàm tính tổng các số lẻ trong dslk đơn vòng các số nguyên.**
- Định nghĩa hàm

```
11. int TongLe(LIST ℓ)
12. {
13.     if(ℓ.pHead==NULL)
14.         return 0;
15.     int s = 0;
16.     NODE*p = ℓ.pHead;
17.     do{
18.         if(p->info%2!=0)
19.             s = s + p->info;
20.         p = p->pNext;
21.     }while(p!=ℓ.pHead);
22.     return s;
23. }
```

8. DUYỆT TUẦN TỰ DSLK ĐƠN VÒNG

- **Ví dụ 2: Định nghĩa hàm xuất dslk đơn vòng các số nguyên.**
- Định nghĩa hàm

```
11. void Output(LIST ℓ)
12. {
13.     if(ℓ.pHead==NULL)
14.         return;
15.     NODE*p = ℓ.pHead;
16.     do{
17.         printf("%4d",p->info)
18.         p = p->pNext;
19.     }while(p!=ℓ.pHead);
20. }
```

9. CHƯƠNG TRÌNH ĐẦU TIÊN DSLK ĐƠN VÒNG

- **Bài toán: Viết chương trình thực hiện các yêu cầu sau:**
  - **Nhập dslk đơn vòng các số nguyên.**
  - **Tính tổng các giá trị trong dslk vòng.**
  - **Xuất dslk vòng.**
- Chương trình

```
#include "stdio.h"
#include "conio.h"
#include "math.h"
#include "string.h"
struct node
{
    int info;
    struct node *pNext;
};
typedef struct node NODE;
```

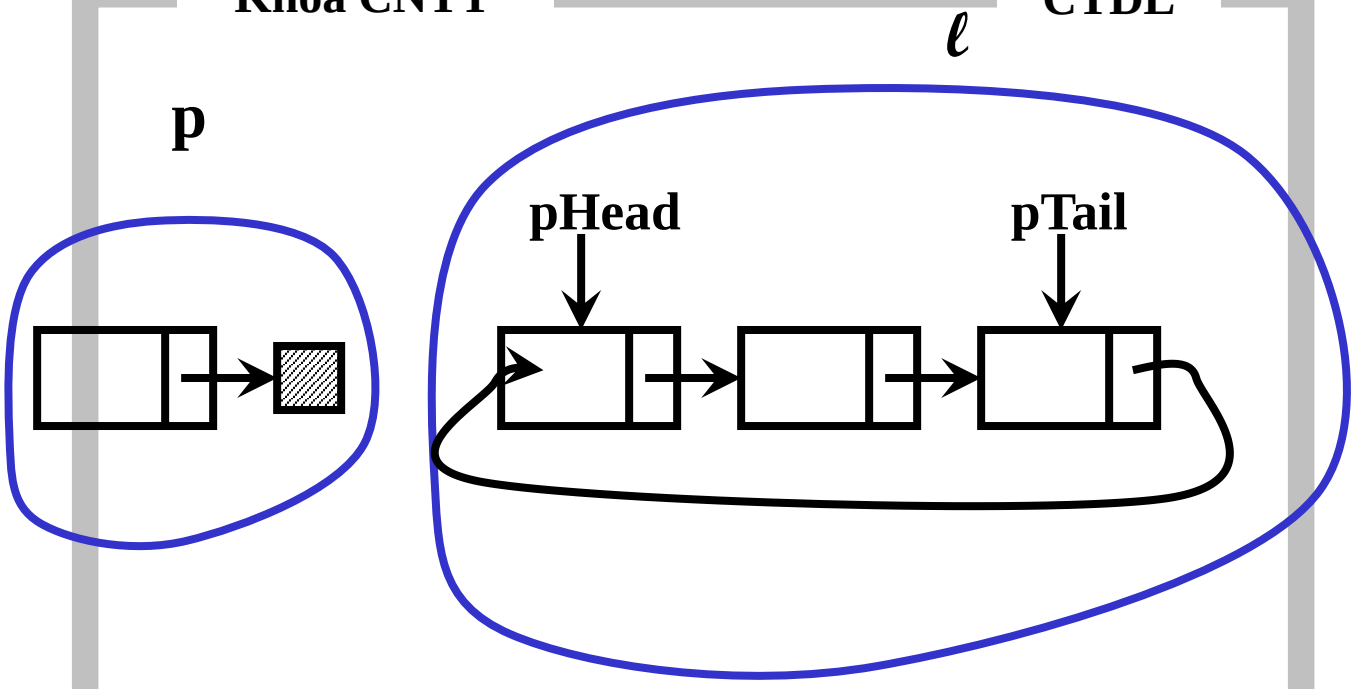
```
11.struct list
12.{
13.|    NODE*pHead;
14.|    NODE*pTail;
15.};
16.typedef struct list LIST;
17.void Init(LIST&);
18.NODE* GetNode(int);
19.void AddHead(LIST&,NODE*);
20.void Input(LIST&);
21.void Output(LIST);
22.int Tong(LIST);
23.void main()
24.{
25.|    LIST lst;
26.|    Input(lst);
27.|    Output(lst);
28.|    int kq = Tong(lst);
29.|    printf("Tong la: %d",kq);
30.}
```

9. CHƯƠNG TRÌNH ĐẦU TIÊN DSLK ĐƠN VÒNG

```
1. void Init(LIST &ℓ)
2. {
3.     ℓ.pHead = NULL;
4.     ℓ.pTail = NULL;
5. }
```

9. CHƯƠNG TRÌNH ĐẦU TIÊN DSLK ĐƠN VÒNG

```
1. NODE* GetNode(int x)
2. {
3.     NODE *p = new NODE;
4.     if (p==NULL)
5.         return NULL;
6.     p->info = x;
7.     p->pNext = p;
8.     return p;
9. }
```



- Định nghĩa hàm:

```
11. void AddHead(LIST&ℓ,NODE*p)
12. {
13.     if(ℓ.pHead==NULL)
14.         ℓ.pHead=ℓ.pTail=p;
15.     else
16.     {
17.         p->pNext = ℓ.pHead;
18.         ℓ.pTail->pNext = p;
19.         ℓ.pHead = p;
20.     }
21. }
```

9. CHƯƠNG TRÌNH ĐẦU TIÊN DSLK ĐƠN VÒNG

```
11. void Input(LIST&ℓ)
12. {
13.     int n;
14.     printf("Nhap n: ");
15.     scanf("%d",&n);
16.     Init(ℓ);
17.     for(int i=1;i<=n;i++)
18.     {
19.         int x;
20.         printf("Nhap so: ");
21.         scanf("%d",&x);
22.         NODE*p = GetNode(x);
23.         if (p!=NULL)
24.             AddHead(ℓ,p);
25.     }
26. }
```

9. CHƯƠNG TRÌNH ĐẦU TIÊN DSLK ĐƠN VÒNG

```
11. void Output(LIST ℓ)
12. {
13.     if(ℓ.pHead==NULL)
14.         return;
15.     NODE*p = ℓ.pHead;
16.     do{
17.         printf("%4d",p->info)
18.         p = p->pNext;
19.     }while(p!=ℓ.pHead);
20. }
```

9. CHƯƠNG TRÌNH ĐẦU TIÊN DSLK ĐƠN VÒNG

```
11. int Tong(LIST ℓ)
12. {
13.     if(ℓ.pHead==NULL)
14.         return 0;
15.     int s = 0;
16.     NODE*p = ℓ.pHead;
17.     do{
18.         s = s + p->info;
19.         p = p->pNext;
20.     }while(p!=ℓ.pHead);
21.     return s;
22. }
```