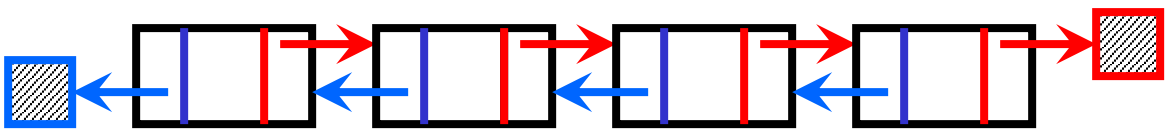
Chương 5
CẤU TRÚC DỮ LIỆU
DANH SÁCH LIÊN KẾT KÉP

TS. Nguyễn Tấn Trần Minh Khang

ThS. Cáp Phạm đình Thăng

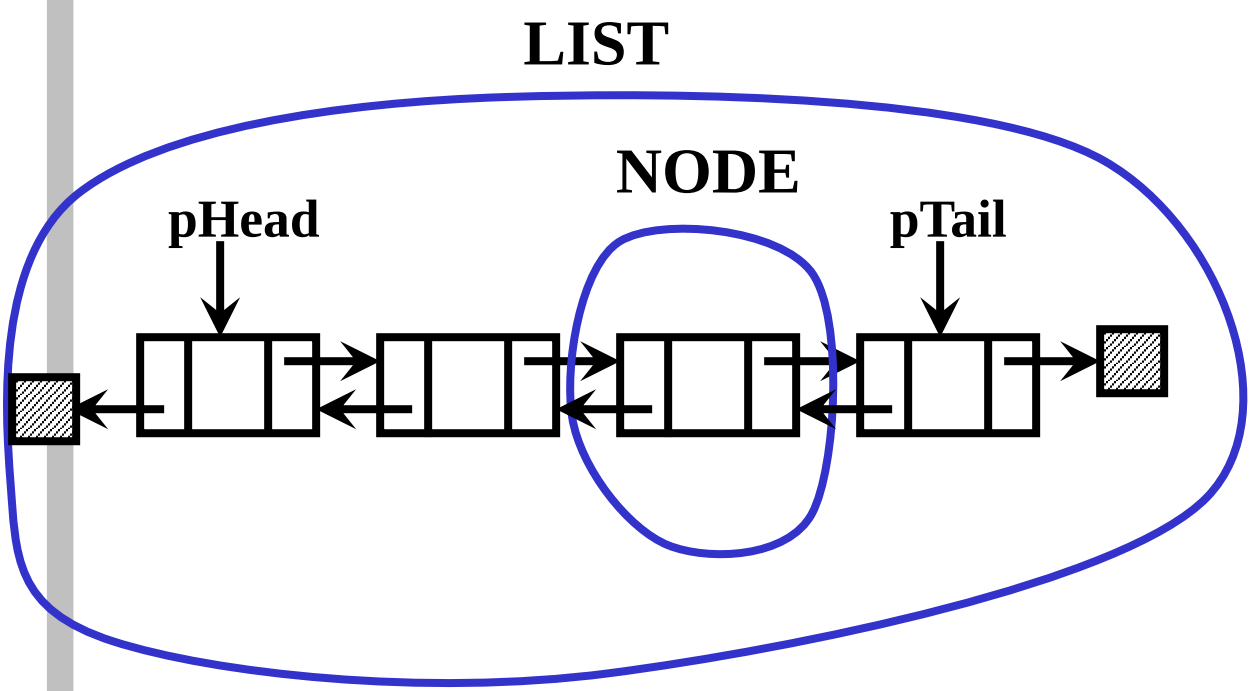
1. HÌNH ẢNH DANH SÁCH LIÊN KẾT KÉP



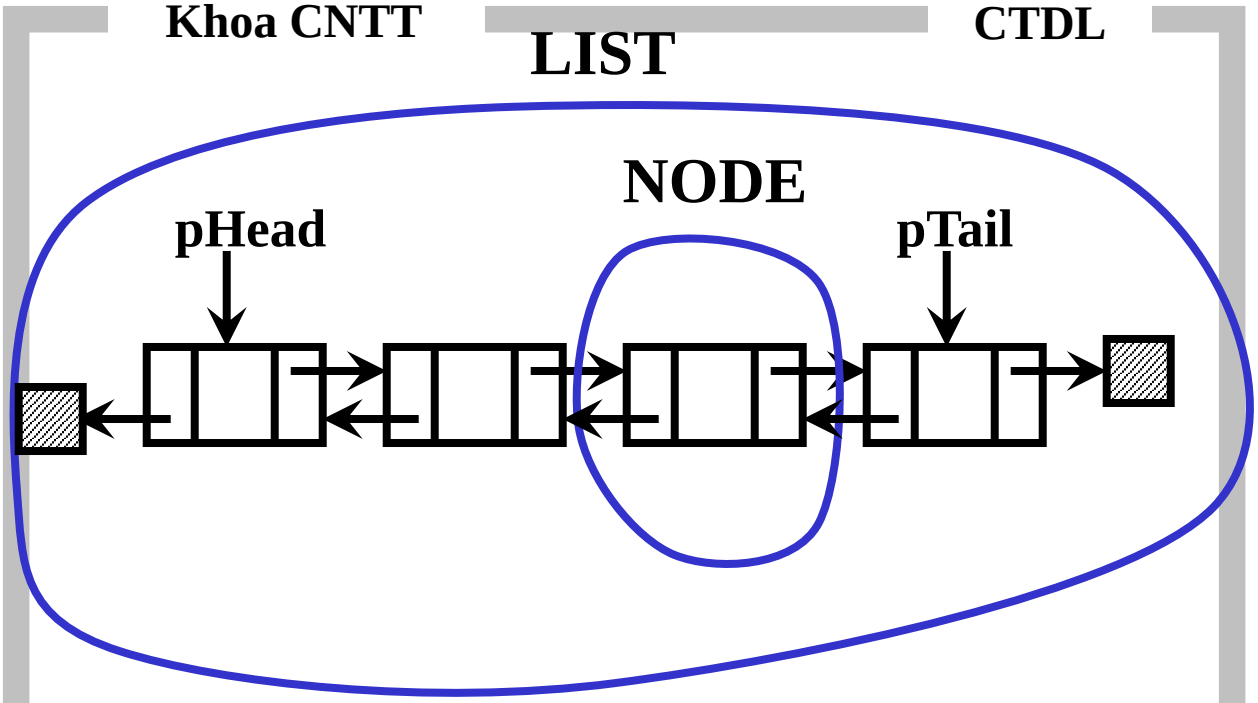
TS. Nguyễn Tấn Trần Minh Khang

ThS. Cáp Phạm đình Thăng

2. CẤU TRÚC DỮ LIỆU DANH SÁCH LIÊN KẾT KÉP



TS. Nguyễn Tấn Trần Minh Khang
ThS. Cáp Phạm đình Thăng




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

2. CẤU TRÚC DỮ LIỆU DANH SÁCH LIÊN KẾT KÉP

– **Ví dụ 1: Hãy khai báo ctdl cho dslk kép các số nguyên.**

```
10. struct node
11. {
12.     int info;
13.     struct node*pNext;
14.     struct node*pPrev;
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 DANH SÁCH LIÊN KẾT KÉP

– **Ví dụ 2: Hãy khai báo ctdl cho dslk kép các số thực.**

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

2. CẤU TRÚC DỮ LIỆU DANH SÁCH LIÊN KẾT KÉP

– **Ví dụ 3: Hãy khai báo ctdl cho dslk kép 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.     struct node*pPrev;
22. };
23. typedef struct node NODE;
24. struct list
25. {
26.     NODE*pHead;
27.     NODE*pTail;
28. };
29. typedef struct list LIST;
```

2. CẤU TRÚC DỮ LIỆU DANH SÁCH LIÊN KẾT KÉP

– **Ví dụ 4: Hãy khai báo ctdl cho dslk kép tọa độ các điểm trong mặt phẳng 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.     struct node*pPrev;
22. };
23. typedef struct node NODE;
24. struct list
25. {
26.     NODE*pHead;
27.     NODE*pTail;
28. };
29. typedef struct list LIST;
```

3. KHỞI TẠO DANH SÁCH LIÊN KẾT KÉP

- **Khái niệm: Khởi tạo danh sách liên kết kép 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 DANH SÁCH LIÊN KẾT KÉP RỖNG

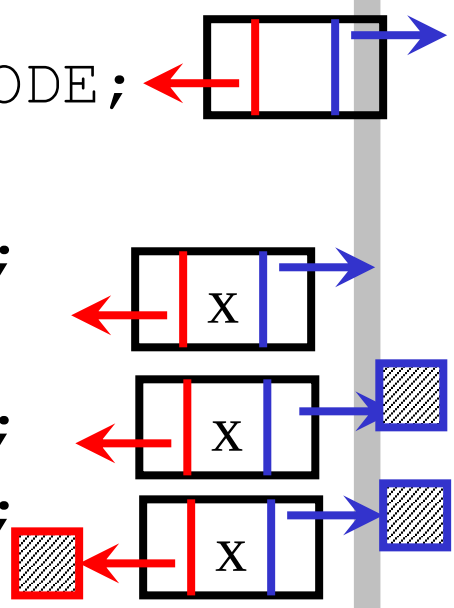
- **Khái niệm: Kiểm tra danh sách liên kết kép rỗng là hàm trả về giá trị 1 khi danh sách rỗng. Trong tình huống danh sách không rỗng thì hàm sẽ trả về giá trị 0.**
- Định nghĩa hàm

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

5. TẠO NODE CHO DANH SÁCH LIÊN KẾT KÉP

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

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



5. TẠO NODE CHO DANH SÁCH LIÊN KẾT KÉP

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

```
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.     p->pPrev = NULL;
19.     return p;
20. }
```

5. TẠO NODE CHO DANH SÁCH LIÊN KẾT KÉP

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

```
11. NODE* GetNode(float x)
12. {
13.     NODE *p = new NODE;
14.     if(p==NULL)
15.         return NULL;
16.     p->info = x;
17.     p->pNext = NULL;
18.     p->pPrev = NULL;
19.     return p;
20. }
```

5. TẠO NODE CHO DANH SÁCH LIÊN KẾT KÉP

- **Ví dụ 3: Định nghĩa hàm tạo một NODE cho dslk kép các phân số để chứa thông tin đã được biết trước.**
- Định nghĩa hàm

```
11. NODE* GetNode(PHANSO x)
12. {
13.     NODE *p = new NODE;
14.     if(p==NULL)
15.         return NULL;
16.     p->info = x;
17.     p->pNext = NULL;
18.     p->pPrev = NULL;
19.     return p;
20. }
```

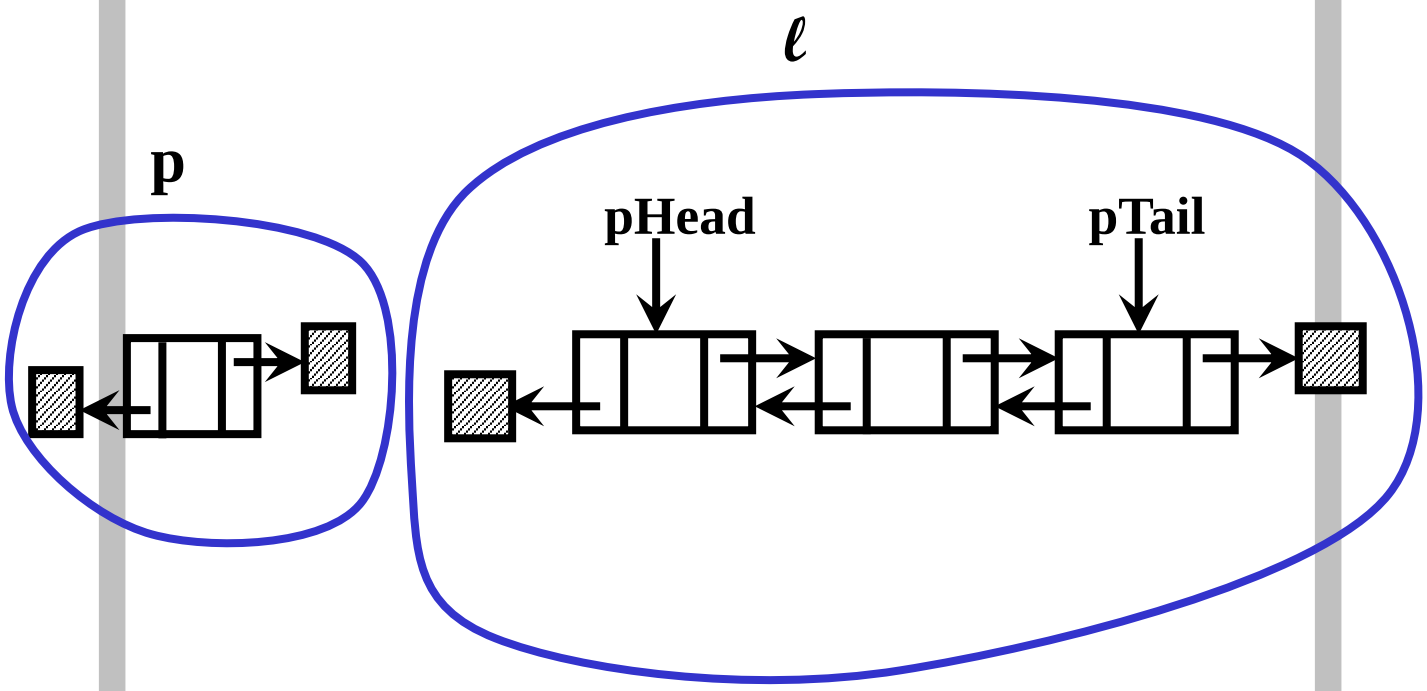
5. TẠO NODE CHO DANH SÁCH LIÊN KẾT KÉP

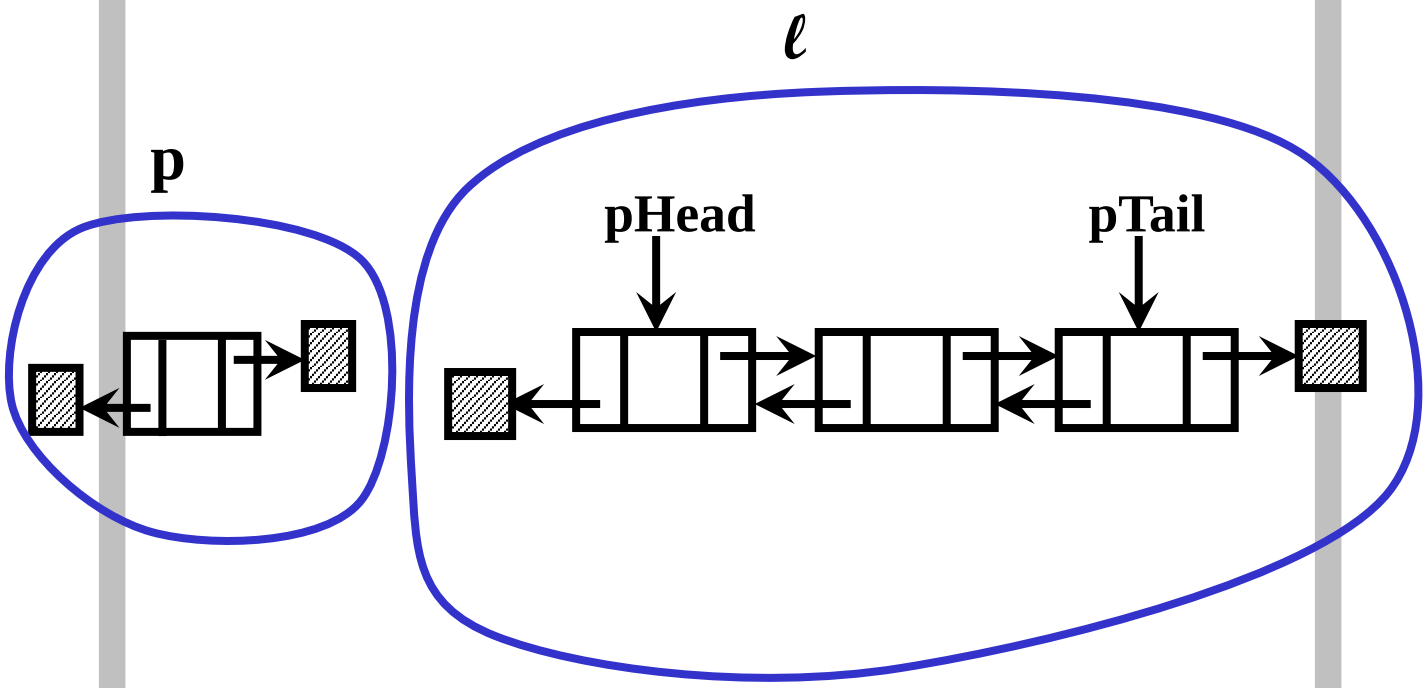
- **Ví dụ 4: Định nghĩa hàm tạo một NODE cho dslk kép tọa độ điểm để chứa thông tin đã được biết trước.**
- Định nghĩa hàm

```
11. NODE* GetNode(DIEM P)
12. {
13.     NODE *p = new NODE;
14.     if(p==NULL)
15.         return NULL;
16.     p->info = P;
17.     p->pNext = NULL;
18.     p->pPrev = NULL;
19.     return p;
20. }
```

6. THÊM MỘT NODE VÀO ĐẦU DANH SÁCH LIÊN KẾT KÉP

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






```
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.pHead->pPrev = p;
19.         ℓ.pHead = p;
20.     }
21. }
```

7. NHẬP TỪ BÀN PHÍM DANH SÁCH LIÊN KẾT KÉP

- **Khái niệm: Nhập từ bàn phím dslk kép là lần lượt nhập các thông tin của từng node trong danh sách.**
- 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 DANH SÁCH LIÊN KẾT KÉP

- **Ví dụ 1: Nhập danh sách liên kết kép 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 nguyen: ");
21.         scanf("%d",&x);
22.         NODE*p=GetNode(x);
23.         if (p!=NULL)
24.             AddHead(ℓ,p);
25.     }
```

7. NHẬP TỪ BÀN PHÍM DANH SÁCH LIÊN KẾT KÉP

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

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

8. DUYỆT TUẦN TỰ DANH SÁCH LIÊN KẾT KÉP

- **Khái niệm: duyệt tuần tự danh sách liên kết kép là thăm qua tất cả các node trong ds mỗi node một lần.**
- Định nghĩa hàm trừu tượng

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

8. DUYỆT TUẦN TỰ DANH SÁCH LIÊN KẾT KÉP

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

```
KDL <Tên Hàm>(LIST ℓ)
{
    ...
    NODE*p = ℓ.pTail;
    while(p!=NULL)
    {
        ...
        p = p->pPrev;
    }
    ...
}
```

8. DUYỆT TUẦN TỰ DANH SÁCH LIÊN KẾT KÉP

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

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

8. DUYỆT TUẦN TỰ DANH SÁCH LIÊN KẾT KÉP

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

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

8. DUYỆT TUẦN TỰ DANH SÁCH LIÊN KẾT KÉP

- **Ví dụ 2: Định nghĩa hàm xuất dslk kép các số nguyên ra màn hình.**
- Định nghĩa hàm

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

9. CHƯƠNG TRÌNH ĐẦU TIÊN DANH SÁCH LIÊN KẾT KÉP

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

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

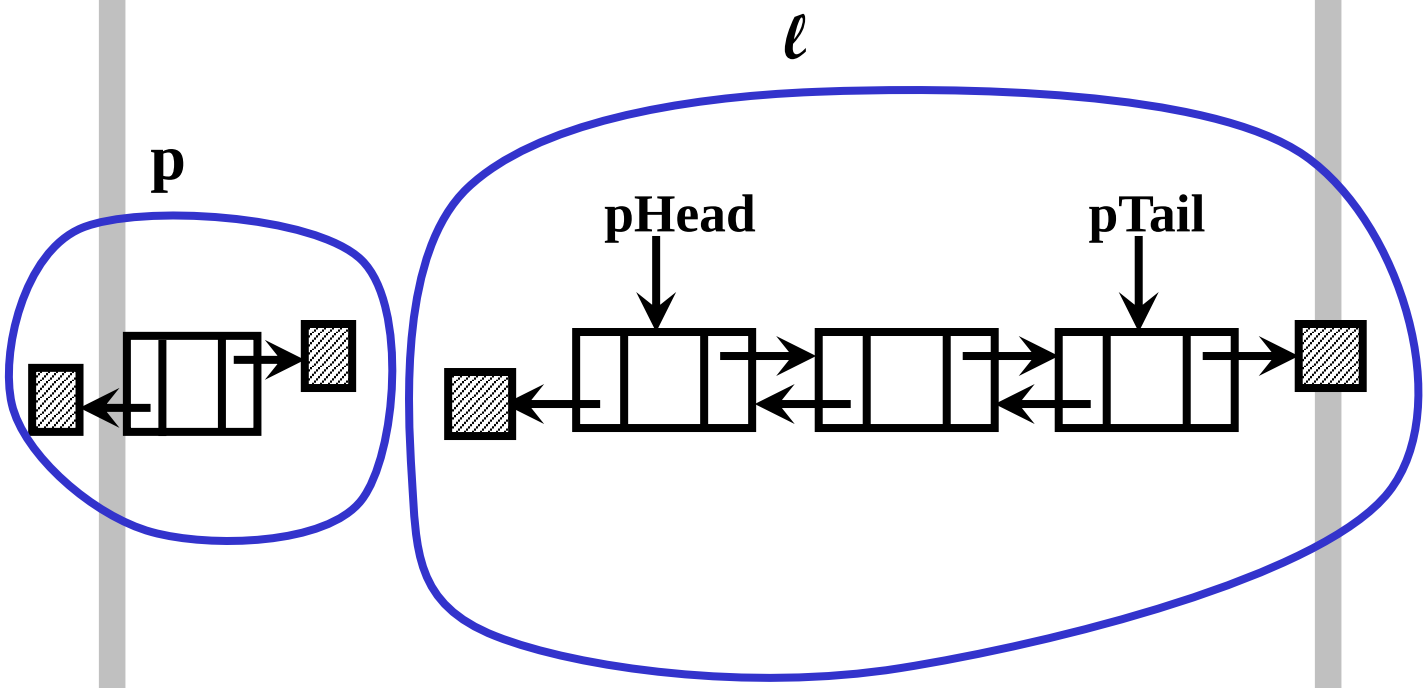
```
10.struct list
11.{
12.|    NODE*pHead;
13.|    NODE*pTail;
14.};
15.typedef struct list LIST;
16.void Init(LIST&);
17.NODE* GetNode(int);
18.void AddHead(LIST&,NODE*);
19.void Input(LIST&);
20.void Output(LIST);
21.int Tong(LIST);
22.void main()
23.{
24.|    LIST lst;
25.|    Input(lst);
26.|  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

9. CHƯƠNG TRÌNH ĐẦU TIÊN DANH SÁCH LIÊN KẾT KÉP

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

9. CHƯƠNG TRÌNH ĐẦU TIÊN DANH SÁCH LIÊN KẾT KÉP

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




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

9. CHƯƠNG TRÌNH ĐẦU TIÊN DANH SÁCH LIÊN KẾT KÉP

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

9. CHƯƠNG TRÌNH ĐẦU TIÊN DANH SÁCH LIÊN KẾT KÉP

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

9. CHƯƠNG TRÌNH ĐẦU TIÊN DANH SÁCH LIÊN KẾT KÉP

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