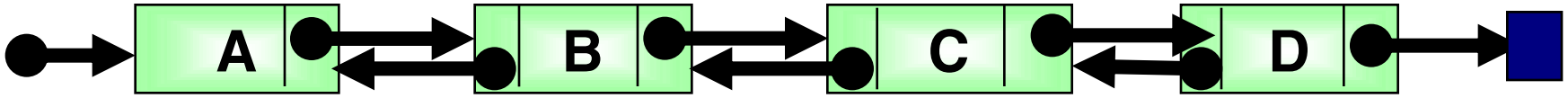
NỘI DUNG

DANH SÁCH LIÊN KẾT kép

Định Nghĩa

- Mỗi phần tử liên kết với phần tử đứng trước và sau nó trong danh sách
- Hình vẽ minh họa danh sách liên kết kép:



Cấu Trúc Dữ Liệu

- *Cấu trúc dữ liệu 1 nút*

```
typedef struct tagDnode
{   Data Info;
    struct tagDnode *pPre;
    struct tagDnode *pNext;
}DNode;
```

- *Cấu trúc List kép*

```
Typedef struct tagDList
{   DNode *pHead;
    DNode *pTail;
}DList;
```

Các Thao Tác Trên List Kép

- Khởi tạo danh sách liên kết kép rỗng
- Tạo 1 nút có thành phần dữ liệu = x
- Chèn 1 phần tử vào danh sách
  - Chèn vào đầu
  - Chèn sau phần tử Q
  - Chèn vào trước phần tử Q
  - Chèn vào cuối danh sách
- Huỷ 1 phần tử trong danh sách
  - Hủy phần tử đầu danh sách
  - Hủy phần tử cuối danh sách
  - Hủy 1 phần tử có khoá bằng x
- Tìm 1 phần tử trong danh sách
- Sắp xếp danh sách

Tạo 1 Danh Sách Rỗng

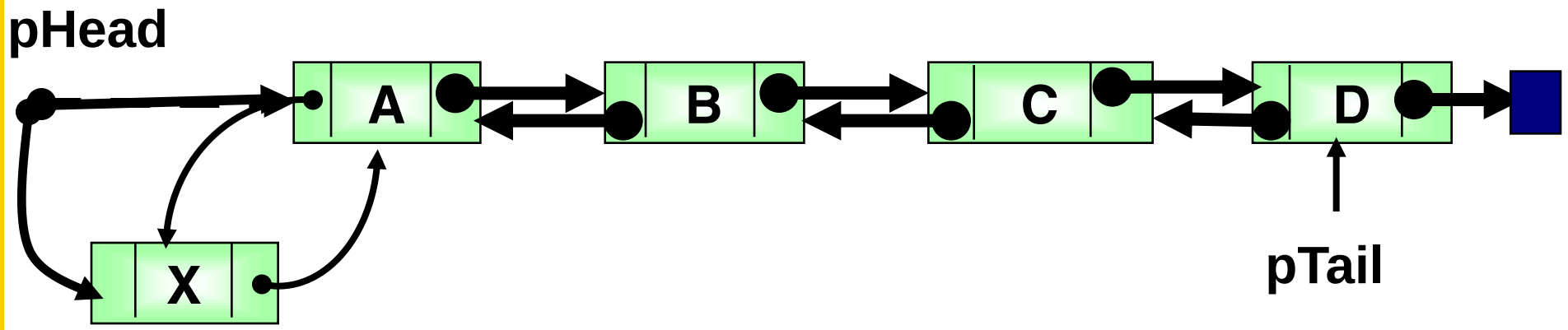
```
void CreateDList(DList &l)
{
        l.DHead=NULL;
        l.DTail=NULL;
}
```


Tạo 1 Nút Có Thành Phần Dữ Liệu = X

```
DNode *CreateDNode(int x)
{  DNode *tam;
   tam=new DNode;
   if(tam==NULL)
   {    printf("khong con du bo nho");
        exit(1);
   }
   else
   {    tam->Info=x;
        tam->pNext=NULL;
        tam->pPre=NULL;
        return tam;
   }
}
```

Thêm 1 Nút Vào Đầu Danh Sách

- Minh họa hình vẽ

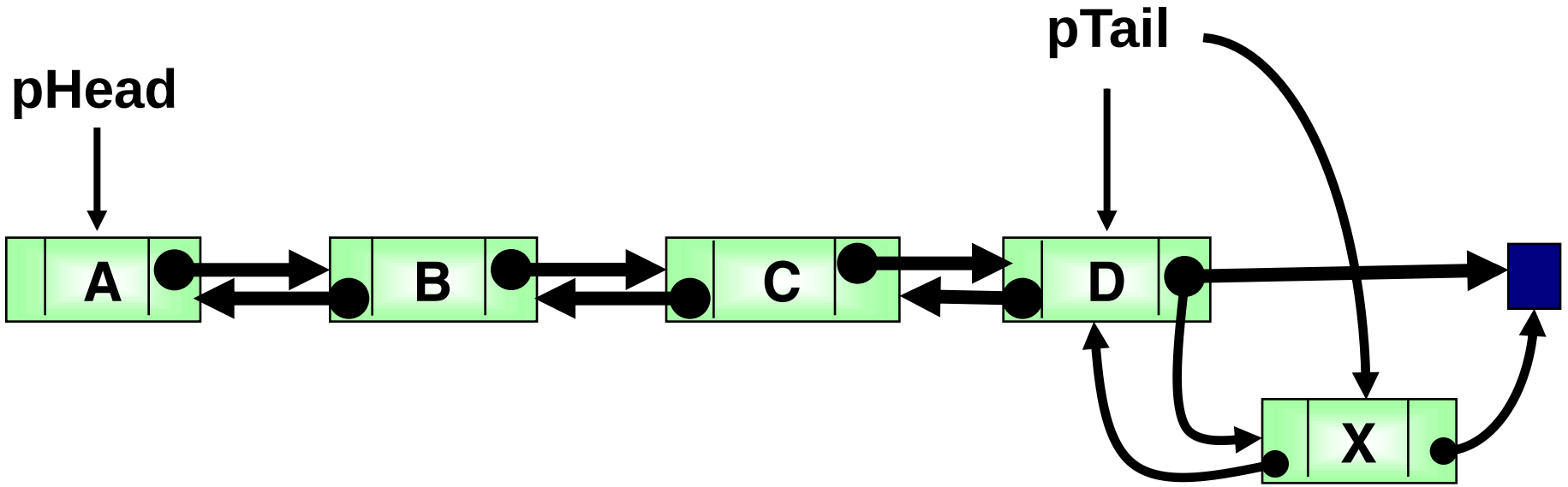


Cài Đặt Thêm 1 Nút Vào Đầu Danh Sách

```
void AddFirst(DList &l, DNode *tam)
{
   if(l.pHead==NULL)//xau rong
   {
        l.pHead=tam;
        l.pTail=l.pHead;
   }
   else
   {
        tam->pNext=l.pHead;
        l.pHead->pPre=tam;
        l.pHead=tam;
   }
}
```

Thêm Vào Cuối Danh Sách

- Minh họa thêm 1 phần tử vào sau danh sách

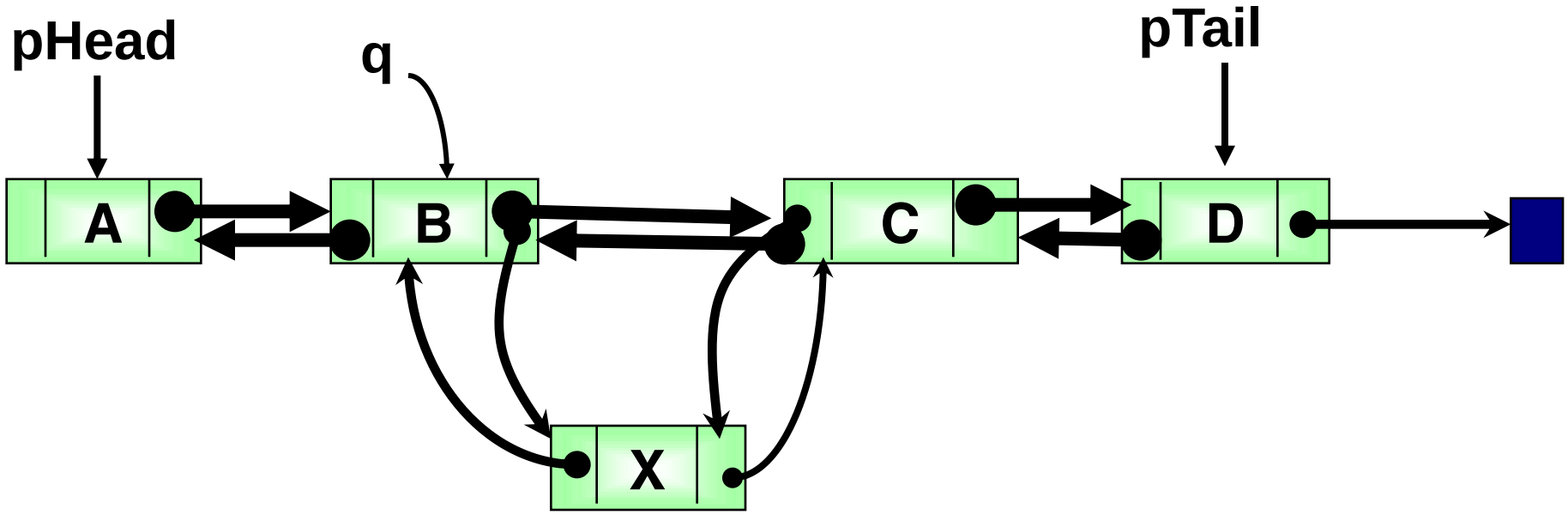


Cài Đặt Thêm 1 Nút Vào Cuối Danh Sách

```
void AddEnd(DList &l,DNode *tam)
{
   if(l.pHead==NULL)
   {
        l.pHead=tam;
        l.pTail=l.pHead;
   }
   else
   {
        tam->pPre=l.pTail;
        l.pTail->pNext=tam;
        tam=l.pTail;
   }
}
```

Thêm Vào Sau Nút Q

- Minh họa thêm nút X vào sau nút q

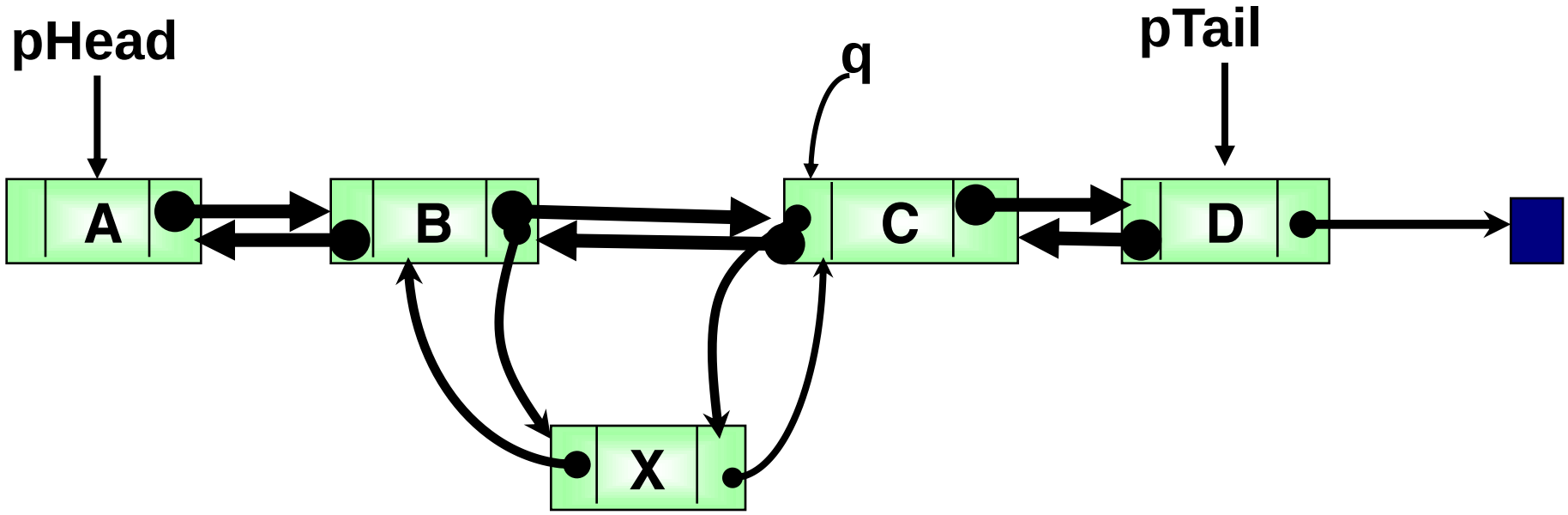


Cài Đặt Thêm 1 Nút Vào Sau Nút Q

```
void AddLastQ(DList &l,DNode *tam, DNode *q)
{
    DNode *p;
    p=q->pNext;
    if(q!=NULL)//them vao duoc
    {
        tam->pNext=p;
        tam->pPre=q;
        q->pNext=tam;
        if(p!=NULL)
            p->pPre=tam;
        if(q==l.pTail) //them vao sau danh sach lien ket.
            l.pTail=tam;
    }
    else
        AddFirst(l,tam);
}
```

Thêm 1 Nút Vào Trước Nút Q

- Minh hoạ thêm 1 nút vào trước nút q



Cài Đặt Thêm 1 Nút Vào Trước Nút Q

```
void AddBeforeQ(DList &l,DNode *tam,DNode *q)
{   DNode *p;
    p=q->pPre;
    if(q!=NULL)
    {
        tam->pNext=q;
        q->pPre=tam;
        tam->pPre=p;
        if(p!=NULL)
            p->pNext=tam;
        if(q==l.pHead)
            l.pHead = tam;
    }
    else
        AddEnd(l,tam);
}
```

Xoá Phần Tử Đầu Danh Sách

```
void DeleteFirst(DList &l)
{
    DNode *p;
    if(l.pHead!=NULL)
    {
        p=l.pHead;
        l.pHead=l.pHead->pNext;
        l.pHead->pPre=NULL;
        delete p;
        if(l.pHead==NULL)
            l.pTail=NULL;
    }
}
```

Xoá 1 Phần Tử Cuối Danh Sách

```
void DeleteEnd(DList &l )
{
    DNode *p;
    if(l.pHead!=NULL) //tuc xau co hon mot phan tu
    {
        p=l.pTail;
        l.pTail=l.pTail->Pre;
        l.pTail->pNext=NULL;
        delete p;
        if(l.pTail==NULL)
            l.pHead=NULL;
    }
}
```

Hủy 1 Nút Sau Nút Q

```
void DeleteLastQ(DList &l,DNode *q)
{
    DNode *p;//luu node dung sau node q
    if(q!=NULL)
    {
        p=q->pNext;
        if(p!=NULL)
        {
            q->pNext=p->pNext;
            if(p==l.pTail)//xoa dung nu't cuoi
                l.pTail=q;
            else      //Nut xoa khong phai nut cuoi
                p->pNext->pPre=q;
            delete p;
        }
    }
    else
        DeleteFirst(l);
}
```

Huỷ 1 Nút Đứng Trước Nút Q

```
void DeleteBeforeQ(DList &l,DNode *q)
{
    DNode *p;
    if(q!=NULL) //tuc ton tai node q
    {
        p=q->pPre;
        if(p!=NULL)
        {
            q->pPre=p->pPre;
            if(p==l.pHead)//p la Node dau cua danh sach
                l.pHead=q;
            else //p khong phai la node dau
                p->pPre->pNext=q;
            delete p;
        }
    }
    else
        DeleteEnd(l);
}
```

Xoá 1 Phần Tử Có Khoá = X

```
int DeleteX(DList &l,int x)
{
    DNode *p;
    DNode *q;
    q=NULL;
    p=l.pHead;
    while(p!=NULL)
    {
        if(p->Info==x)
            break;
        q=p;//q la Node co truong Info = x
        p=p->pNext;
    }
    if(q==NULL) return 0;//khong tim thay Node nao co truong Info =x
    if(q!=NULL)
        DeleteLastQ(l,q);
    else
        DeleteFirst(l);
    return 1;
}
```

Sắp Xếp

```
void DoiChoTrucTiep(DList &l)
{  DNode *p,*q;
   p=l.pHead;
   while(p!=l.pTail)
   {
        q=p->pNext;
        while(q!=NULL)
        {
             if(p->Info>q->Info)
                  HV(p,q);
             q=q->pNext;
        }
        p=p->pNext;
   }}
```