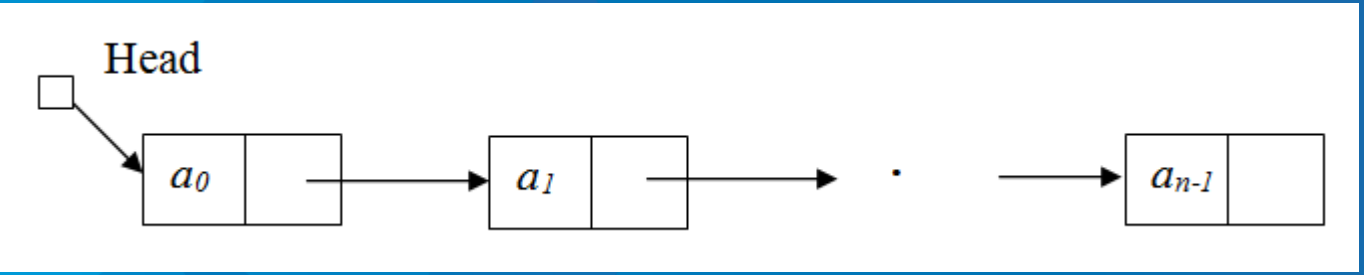
Khai báo



Khai báo

```
struct typeOfNode
{
    typeOfData data;
    struct typeOfNode *next;
};
typedef struct typeOfNode *nodePtr;
```

Hàm khởi tạo

```
void initialize(NodePtr *L)
      {
      *L = NULL;
      }
```

Hàm kiểm tra danh sách rỗng

```
int isempty(NodePtr L)
      {
      return ((L == NULL)?True:False);
      }
```

Tìm kiếm

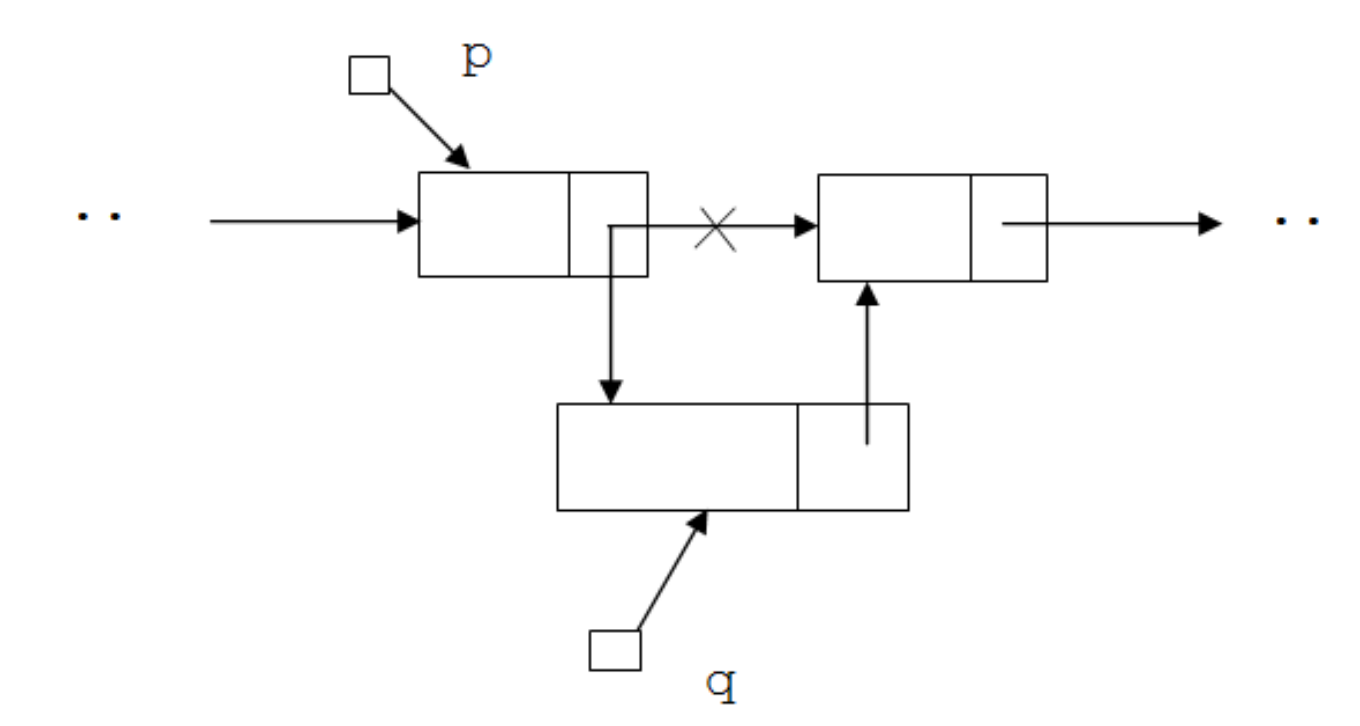
- Trả lại con trỏ trỏ vào phần tử cần tìm hoặc NULL

```
NodePtr search(int x, NodePtr L)
{
 if (L == NULL)
   return NULL;
 else
    {
     NodePtr q;
     q = L;
     while (q!= NULL && q->data!=x)
          q = q->next;
     return q;
    }
}
```

Duyệt danh sách

```
void traverse(NodePtr L)
{
 if (L==NULL)
   printf("\n Danh sach rong!");
 else
   {
    NodePtr q;
    q = L;
    while (q!=NULL)
         {
          Visit(q);
          q=q->next;
         }
   }
}
```

Chèn

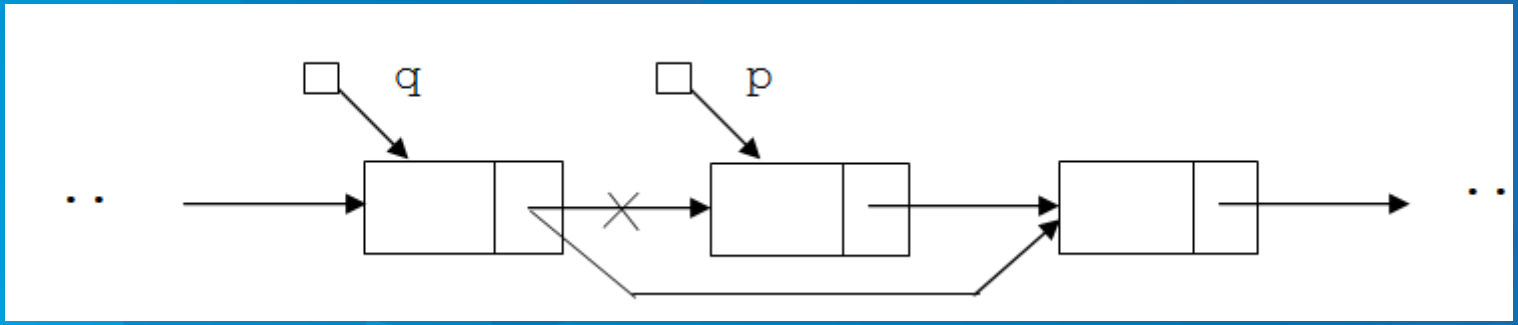


Chèn

```
void  InsertAfter(int x, NodePtr  p,  NodePtr *L)
{
 NodePtr q;

 if (p==NULL)
   printf("\n Khong them duoc!");
 else
      {
       q = (NodePtr)malloc(sizeof(struct node));
       q->data = x;
       q->next = p->next;
       p->next = q;
      }
}
```

Loại bỏ



Loại bỏ

```
void   DelList(NodePtr p, NodePtr *L)
{
 NodePtr q;
 if (p==NULL)
   printf("\n Phan tu can xoa ko ton tai trong ds!");
 else
     if (isempty(*L))
       printf("\n Danh sach rong!");
     else
        {
         q = *L;//xac dinh q la nut dung truoc nut p
         while (q!=NULL && q->next != p)
             q=q->next;
         q->next = p->next;
         free(p);
        }
}
```