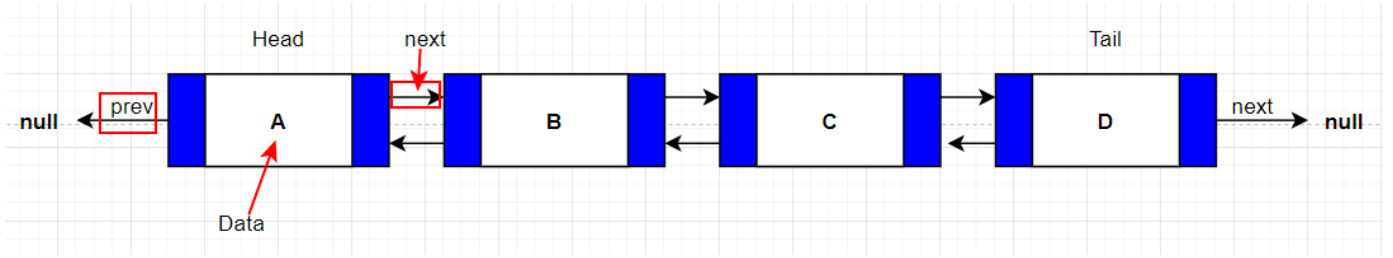
Bài 10.5: Danh sách liên kết đôi

- ✓ Tạo danh sách liên kết đôi
- ✓ Thêm node vào đầu danh sách
- ✓ Thêm node vào cuối danh sách
- ✓ Thêm node vào sau node x
- ✓ Duyệt danh sách liên kết
- ✓ Ví dụ và bài tập thực hành

Tạo danh sách liên kết đôi

✓ Ta thực hiện hai bước: tạo node và tạo danh sách liên kết đôi.



✓ Ví dụ sau tạo node trong ngôn ngữ lập trình C#:

```
class Node<T>
{
    3 references
    public T Data { get; set; } // dữ liệu của node
    9 references
    public Node<T> Next { get; set; } // địa chỉ node tiếp theo
    2 references
    public Node<T> Prev { get; set; } // địa chỉ node liền trước
    0 references
    public Node()
    {
        Next = null;
        Prev = null;
    }
    3 references
    public Node(T data)
    {
        Data = data;
        Next = null;
        Prev = null;
    }
}
```

Tạo danh sách liên kết đôi

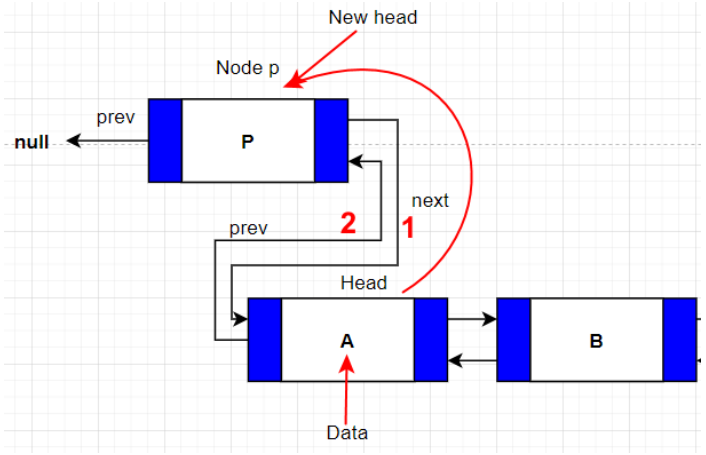
- ✓ Tiếp theo ta tạo danh sách liên kết đôi với các tùy chọn:
- ✓ Tạo danh sách liên kết đôi chỉ gồm 1 node first.
- ✓ Tạo danh sách liên kết đôi bao gồm 2 node first, last.
- ✓ Ví dụ sau tạo danh sách liên kết đôi gồm hai node first, last:

```
class LinkedList<T>
{
    8 references
    public Node<T> First { get; set; }
    7 references
    public Node<T> Last { get; set; }
    1 reference
    public LinkedList()...
    // thêm node vào đầu danh sách lk
    5 references
    public void AddFirst(T data)...
    // thêm node vào cuối danh sách lk
    3 references
    public void AddLast(T data)...
    // thêm node vào sau node x danh sách lk
    2 references
    public void AddAfterX(T data, T x)...
    // kiểm tra danh sách rỗng
    3 references
    public bool IsEmpty()...
    // hiển thị danh sách các node hiện có
    2 references
    public void ShowNodes()...
}
```

Thêm node vào đầu DSLK

- ✓ Các bước thực hiện để chèn node p vào đầu danh sách liên kết:
- ✓ Kiểm tra xem danh sách có rỗng hay không.
- ✓ Nếu danh sách rỗng, ta gán **first = last = p**.
- ✓ Nếu danh sách không rỗng:
  - ✓ Gán first cho p->next: **p->next = first;**
  - ✓ Gán p cho first->prev: **first->prev = p;**
  - ✓ Cập nhật lại first là p: **first = p;**

Code mẫu

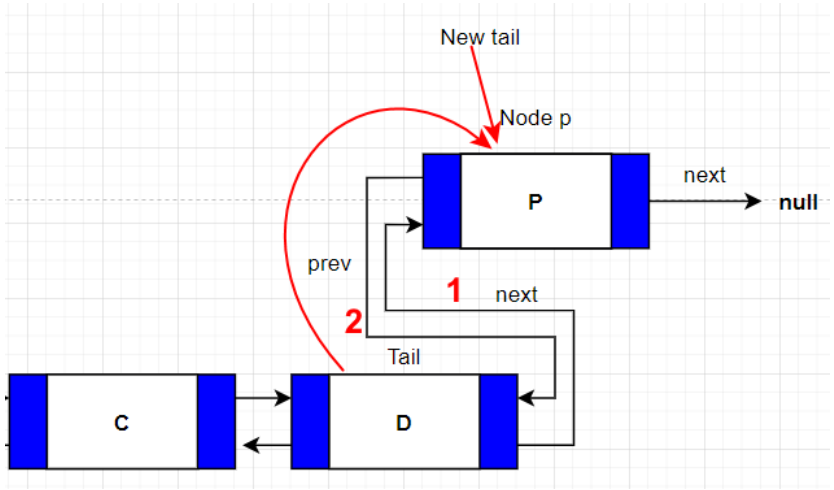



```
// thêm node vào đầu danh sách lk
5 references
public void AddFirst(T data)
{
    var p = new Node<T>(data); // tạo node cần chèn
    if (IsEmpty()) // nếu danh sách rỗng
    {
        First = p; // gán p cho first
        Last = p; // gán p cho last
    }
    else // nếu danh sách không rỗng
    {
        p.Next = First; // cập nhật next của p
        First.Prev = p; // cập nhật prev của first
        First = p; // cập nhật lại node first mới
    }
}
```

Thêm node vào cuối DSLK

Giả sử ta muốn thêm node p vào cuối danh sách liên kết hiện tại:

- ✓ Nếu danh sách rỗng, ta gán **first = last = p**.
- ✓ Nếu danh sách không rỗng, ta thực hiện:
  - ✓ Gán p làm next của last: **last->next = p;**
  - ✓ Gán last cho prev của p: **p->prev = last;**
  - ✓ Cập nhật lại node last mới: **last = p;**



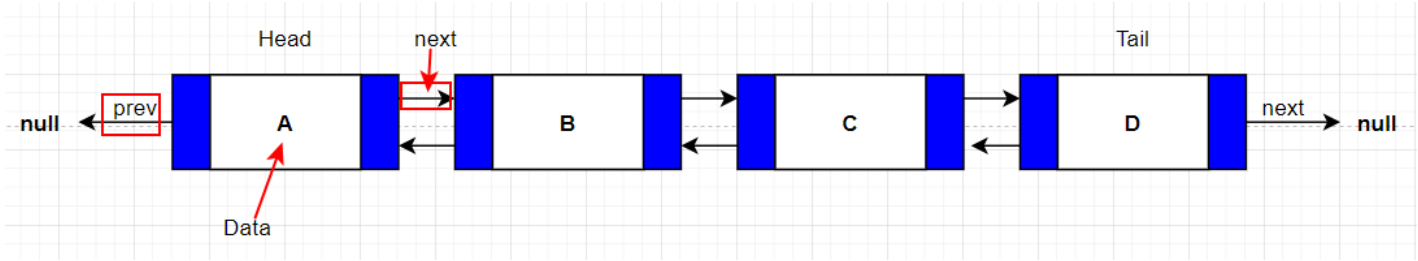
Code mẫu

```
// thêm node vào cuối danh sách lk
3 references
public void AddLast(T data)
{
    var p = new Node<T>(data);
    if (IsEmpty()) // nếu danh sách rỗng
    {
        First = p; // first và last cùng tham chiếu tới p
        Last = p;
    }
    else // nếu danh sách không rỗng
    {
        Last.Next = p; // cập nhật next của last
        p.Prev = Last; // cập nhật prev của p
        Last = p; // cập nhật lại node last mới
    }
}
```

Thêm node vào sau node x

Giả sử ta muốn thêm node p sau node có giá trị B:

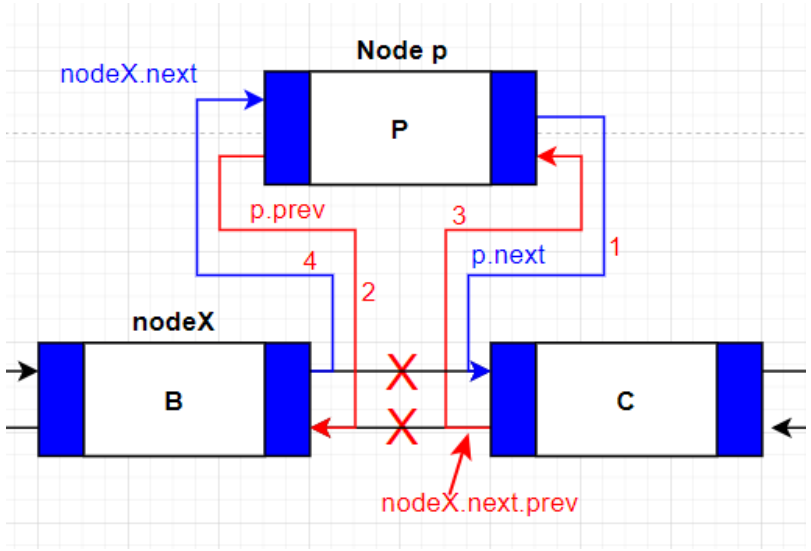
- ✓ Tìm node chứa data bằng B gọi là nodeX.
- ✓ Nếu tìm thấy:
  - ✓ Gán node next của nodeX cho next của p: **p->next = nodeX->next;**
  - ✓ Gán nodeX cho prev của p: **p->prev = nodeX;**
  - ✓ Gán p cho prev của nodeX->next nếu nodeX->next khác null: **nodeX->next->prev = p;**
  - ✓ Cập nhật next của nodeX: **nodeX->next = p;**
- ✓ Nếu không tìm thấy node cần tìm, thông báo ra màn hình và kết thúc.



Code mẫu

```
// thêm node vào sau node x danh sách lk
2 references
public void AddAfterX(T data, T x)
{
    if (IsEmpty()) // nếu danh sách rỗng
    {
        AddFirst(data); // chèn vào đầu
    }
    else
    {
        var currentNode = First;
        while (currentNode != null) // tìm node có giá trị x
        {
            if (currentNode.Data.Equals(x)) // nếu tìm thấy
            {
                if (currentNode == Last) // node hiện tại là node cuối
                {
                    AddLast(data); // chèn vào cuối
                }
                else
                {
                    var p = new Node<T>(data); // tạo node p
                    p.Next = currentNode.Next; // cập nhật next của p
                    p.Prev = currentNode; // cập nhật prev của p
                    currentNode.Next.Prev = p; // cập nhật node prev của node sau p
                    currentNode.Next = p; // cập nhật node next của currentNode
                }
                break;
            }
            currentNode = currentNode.Next;
        }
    }
}
```

Hình minh họa



Duyệt danh sách liên kết

- ✓ Có thể duyệt theo chiều xuôi từ đầu danh sách tới cuối danh sách.
- ✓ Với cách này, khai báo node p và khởi tạo bằng first: p = first;
- ✓ Tiếp theo, lặp chừng nào p chưa null:
  - ✓ Xuất ra giá trị của p->data;
  - ✓ Cập nhật p: p = p->next;
- ✓ Hoặc duyệt ngược từ cuối danh sách lên đầu.
- ✓ Với cách này, ta khai báo node p và khởi tạo bằng last: p = last;
- ✓ Sau đó lặp chừng nào p chưa null:
  - ✓ Xuất ra giá trị của p->data;
  - ✓ Cập nhật p: p = p->prev;

Duyệt danh sách liên kết

```
// hiển thị danh sách các node hiện có
2 references
public void ShowNodes()
{
    var x = First;
    while (x != null)
    {
        Console.Write($"{x.Data} -> ");
        x = x.Next;
    }
    Console.WriteLine("null");
}

// hiển thị danh sách node theo thứ tự ngược lại
1 reference
public void ShowNodesReverse()
{
    var x = Last;
    while (x != null)
    {
        Console.Write($"{x.Data} <- ");
        x = x.Prev;
    }
    Console.WriteLine("null");
}
```

Test chương trình

```
Console.OutputEncoding = Encoding.UTF8;
LinkedList<string> list = new LinkedList<string>();
// chèn đầu DSLK
list.AddFirst("One");
list.AddFirst("Two");
list.AddFirst("Three");
list.AddFirst("Four");
// chèn cuối DSLK
list.AddLast("Five");
list.AddLast("Six");
Console.WriteLine("==> Các node trong danh sách liên kết trước khi chèn: ");
list.ShowNodes();
// chèn sau giá trị x
list.AddAfterX("Nine", "One");
list.AddAfterX("Nine", "Six");
Console.WriteLine("==> Các node trong danh sách liên kết sau khi chèn: ");
list.ShowNodes();
list.ShowNodesReverse();
```

```
C:\WINDOWS\system32\cmd.exe
==> Các node trong danh sách liên kết trước khi chèn:
Four -> Three -> Two -> One -> Five -> Six -> null
==> Các node trong danh sách liên kết sau khi chèn:
Four -> Three -> Two -> One -> Nine -> Five -> Six -> Nine -> null
Nine -> Six -> Five -> Nine -> One -> Two -> Three -> Four -> null
```

Nội dung tiếp theo

Tìm hiểu về stack