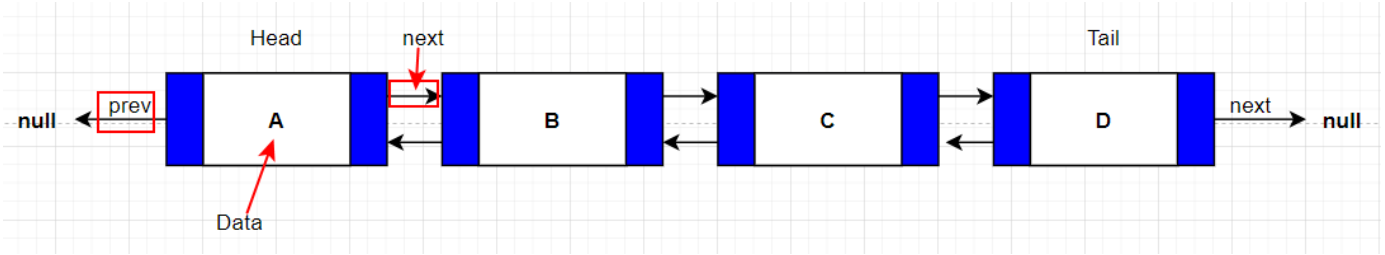
Bài 3.3: Chèn node vào danh sách

- ✓ 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.
- Để tạo node ta sử dụng kiểu dữ liệu người dùng tự định nghĩa như struct(trong C) hoặc struct/class trong C++.



- Ví dụ sau tạo node trong ngôn ngữ lập trình C++:

```
template<class T> class Node {
public:
    T data;          // phần dữ liệu của một node
    Node<T>* next;   // con trỏ next chứa địa chỉ node kế tiếp
    Node<T>* prev;   // con trỏ prev chứa địa chỉ node phía trước
    Node(T data) {
        this->next = null;
        this->prev = null;
        this->data = data; // gán giá trị
    }
};
```

Tạo danh sách liên kết đôi

Tiếp theo ta tạo danh sách liên kết đôi với các tùy chọn:

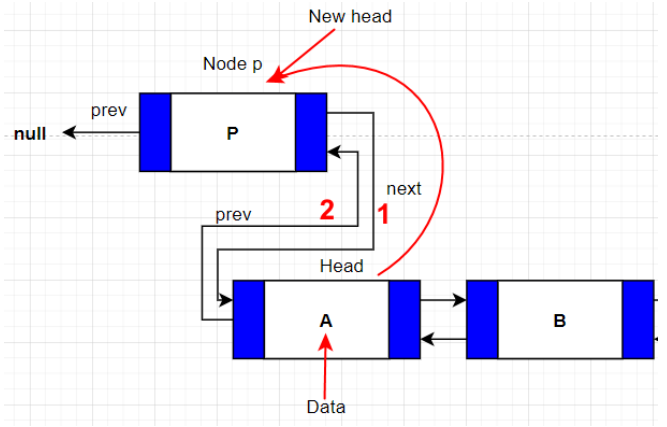
- Tạo danh sách liên kết đôi chỉ gồm 1 node head.
- Tạo danh sách liên kết đôi bao gồm 2 node head, tail.
- Ví dụ sau tạo danh sách liên kết đôi gồm hai node head, tail:

```
template<class T> class DoublyLinkedList {
    Node<T>* head; // node đầu tiên trong danh sách
    Node<T>* tail; // node cuối cùng trong danh sách
public:
    DoublyLinkedList();
    void add(T data);
    void addTail(T data);
    void addAfterX(T data, T x);
    bool isEmpty();
    void showNodes();
};
```

Thêm node vào đầu danh sách liên kết

- 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 **head = tail = p**.
- Nếu danh sách không rỗng:
  - Gán head cho p->next: **p->next = head;**
  - Gán p cho head->prev: **head->prev = p;**
  - Cập nhật lại head là p: **head = p;**

Code mẫu

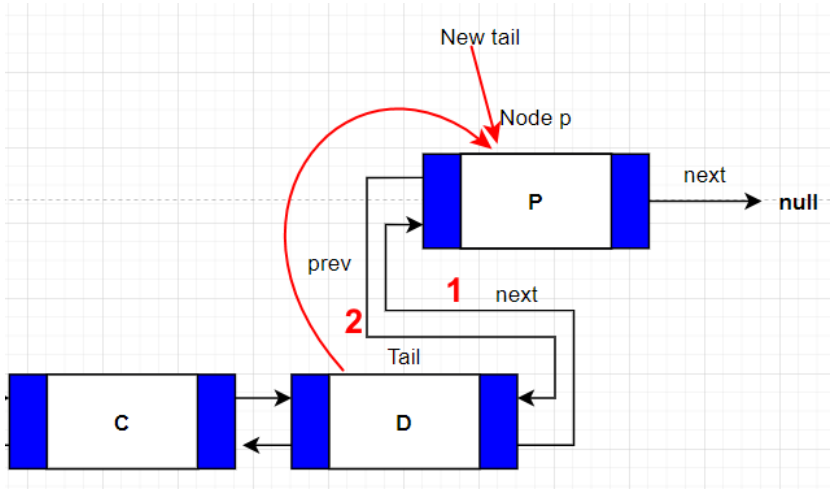



```
void add(T data) {
    Node<T>* p = new Node<T>(data);
    if (isEmpty()) { // nếu danh sách rỗng
        head = tail = p; // gán giá trị cho head, tail
    }
    else { // nếu danh sách không rỗng
        p->next = head; // cập nhật next của p
        head->prev = p; // cập nhật prev của head
        head = p; // cập nhật head
    }
}
```

Thêm node vào cuối danh sách

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

- Nếu danh sách rỗng, ta gán **head = tail = p**.
- Nếu danh sách không rỗng, ta thực hiện:
  - Gán p làm next của tail: **tail->next = p;**
  - Gán tail cho prev của p: **p->prev = tail;**
  - Cập nhật lại tail: **tail = p;**



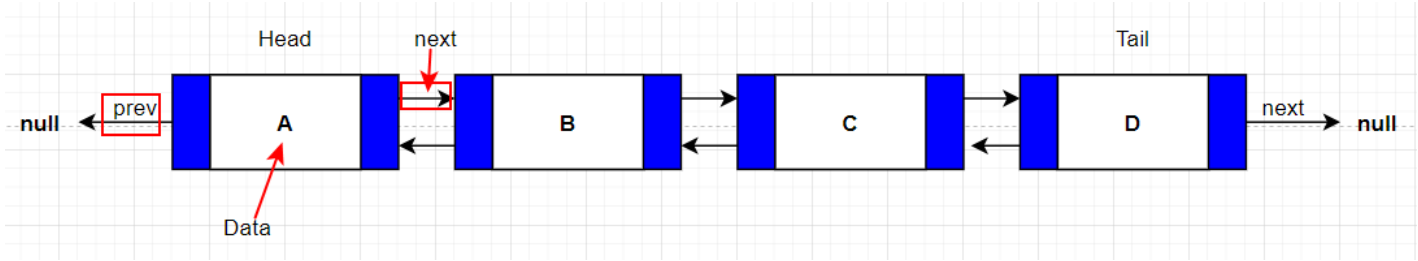
Code mẫu

```
void addTail(T data) {
    Node<T>* p = new Node<T>(data);
    if (isEmpty()) { // nếu danh sách rỗng
        head = tail = p; // gán giá trị cho head, tail
    }
    else { // nếu danh sách không rỗng
        tail->next = p; // cập nhật next của tail
        p->prev = tail; // cập nhật prev của p
        tail = p; // cập nhật tail mới
    }
}
```

Thêm node vào sau node x

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

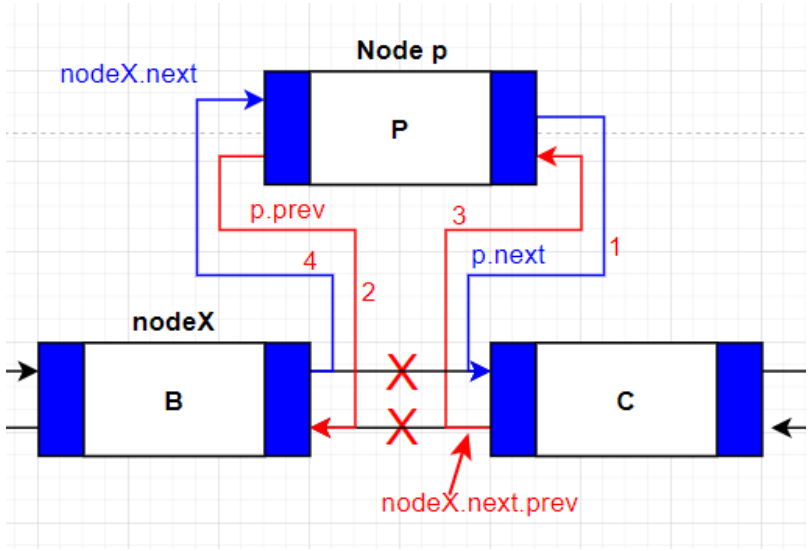
- Tìm node chứa data bằng B gọi là nodeX.
- Nếu tìm thấy:
  - Gán node next của nodeX cho next của p: **p->next = nodeX->next;**
  - Gán nodeX cho prev của p: **p->prev = nodeX;**
  - Gán p cho prev của nodeX->next nếu nodeX->next khác null: **nodeX->next->prev = p;**
  - Cập nhật next của nodeX: **nodeX->next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



Code mẫu

```
void addAfterX(T data, T x) {
    Node<T>* p = new Node<T>(data); // tạo node p
    Node<T>* nodeX = head; // bắt đầu từ head
    while (nodeX != nullptr) { // chừng nào nodeX chưa null
        if (nodeX->data == x) { // nếu tìm thấy node cần tìm
            break; // dừng việc tìm kiếm
        }
        nodeX = nodeX->next; // tiếp tục tìm
    }
    if (nodeX != nullptr) { // nếu tìm thấy
        p->next = nodeX->next; // cập nhật next của p
        p->prev = nodeX; // cập nhật prev của p
        if (nodeX->next != nullptr) { // nếu next của nodeX khác null
            nodeX->next->prev = p; // cập nhật prev của next của nodeX
        }
        nodeX->next = p; // cập nhật next của nodeX
    }
}
```

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 head: p = head;
- 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 tail: p = tail;
- 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

```
void showNodes() { // duyệt xuôi
    Node<T>* p = head; // bắt đầu từ đầu danh sách
    while (p != nullptr) {
        cout << p->data << " ";
        p = p->next; // tiến về phía phải
    }
}

void showNodesRev() { // duyệt ngược
    Node<T>* p = tail; // từ cuối danh sách
    while (p != nullptr) {
        cout << p->data << " ";
        p = p->prev; // tiến về phía trái
    }
}
```

Nội dung tiếp theo

Cập nhật dữ liệu cho một node