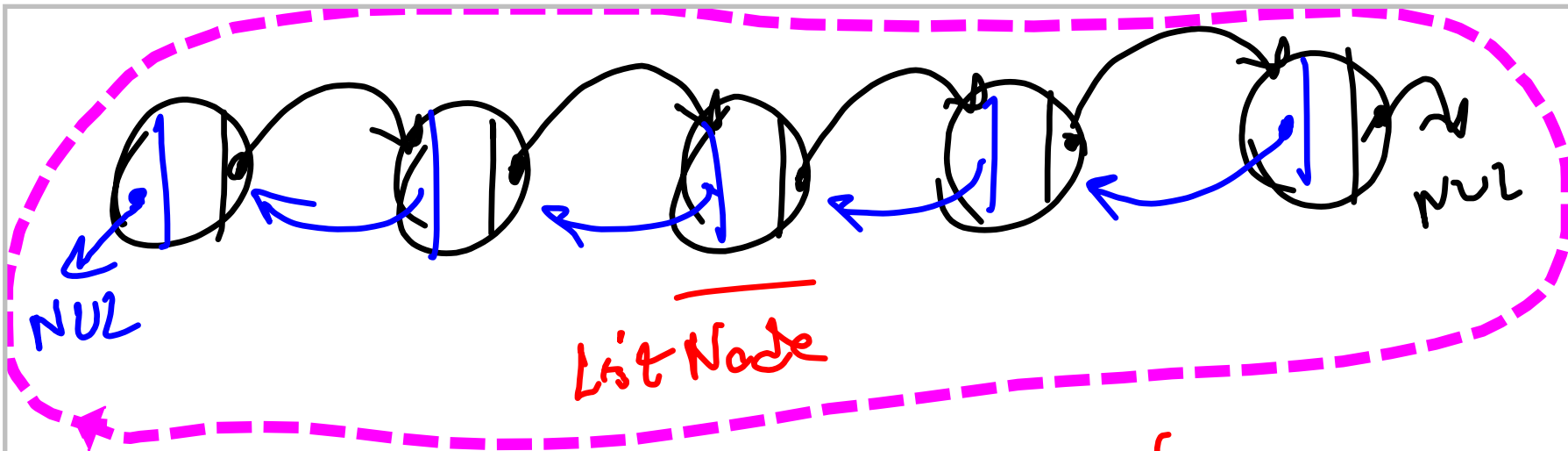


Doubly Linked

Lists

MP3 : by midnight

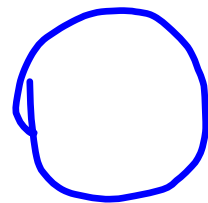
Due : 2/26 NOON



```

template <class X> List Node {
    ListNode <X> * next;
    ListNode <X> * prev;
} X data;

```



```

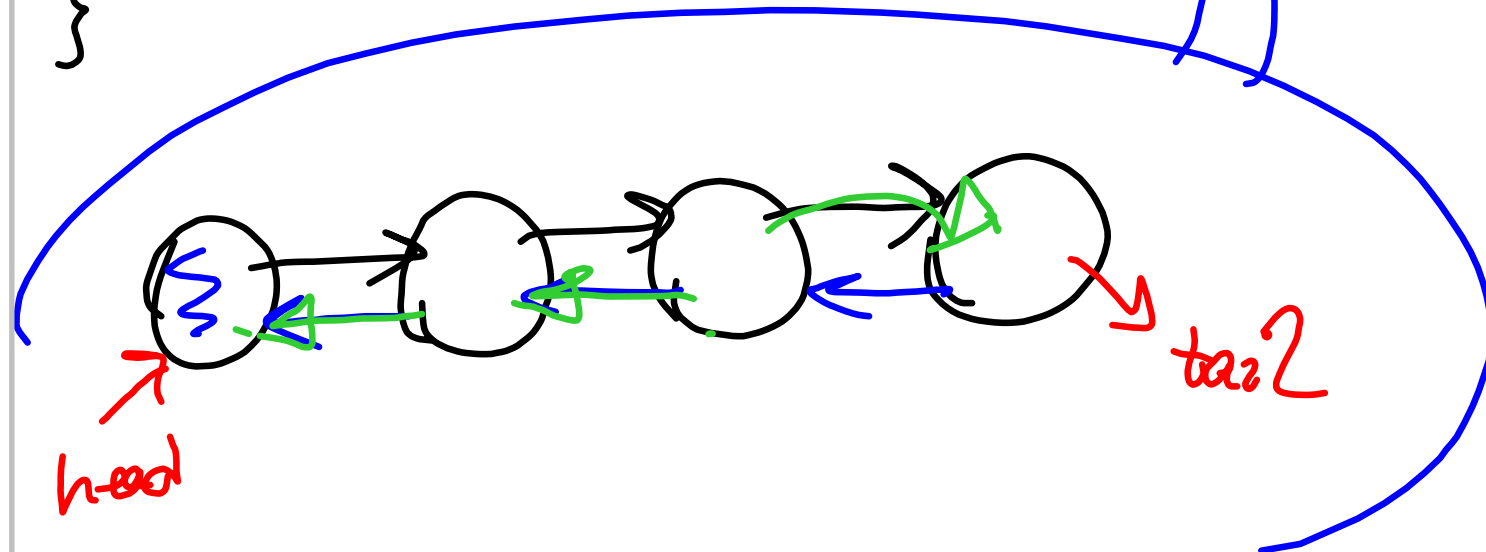
template <class X> List {
  ListNode<X> * head;
  ListNode<X> * tail;

```

```

    _____ } methods
    _____
    _____
    _____ }

```



```

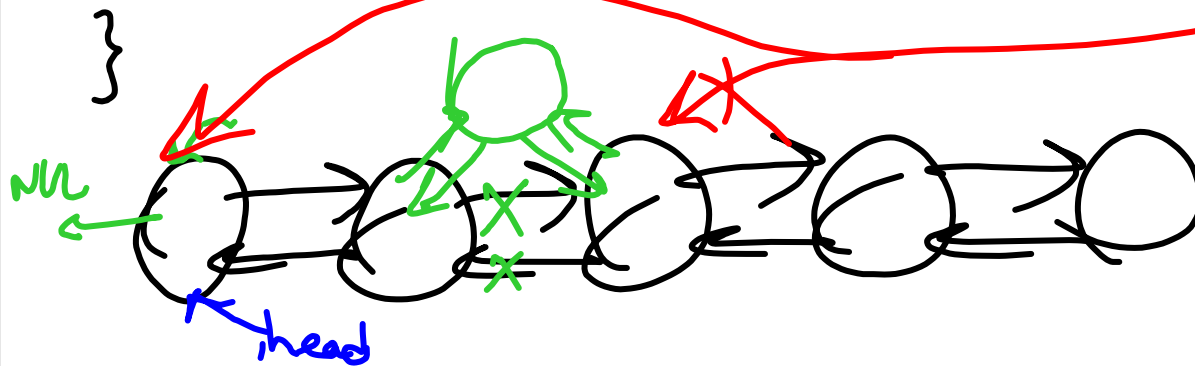
template <class X> List {
  ListNode<X> * head;
  ListNode<X> * tail;
  const ~
  ;
}

```

```

void insertAfter (ListNode<X> * node);
void insertBefore (ListNode<X> * node);

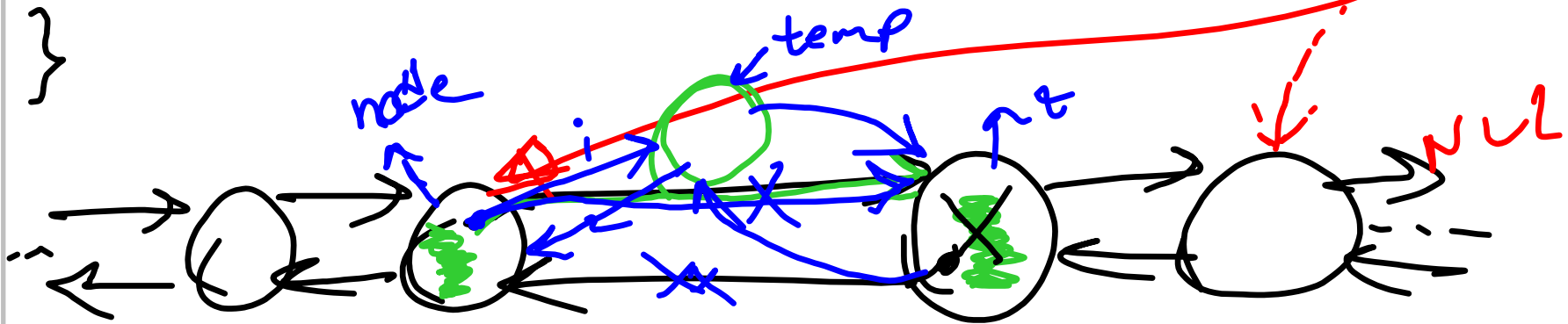
```



```

void List<X>::insertAfter (ListNode<X> *
                        node ) {
    ListNode<X> * temp = new ListNode<X>;
    ListNode<X> * t = node -> next;
    node -> next = temp;
    temp -> prev = node;
    temp -> next = t;
    t -> prev = temp;
}

```

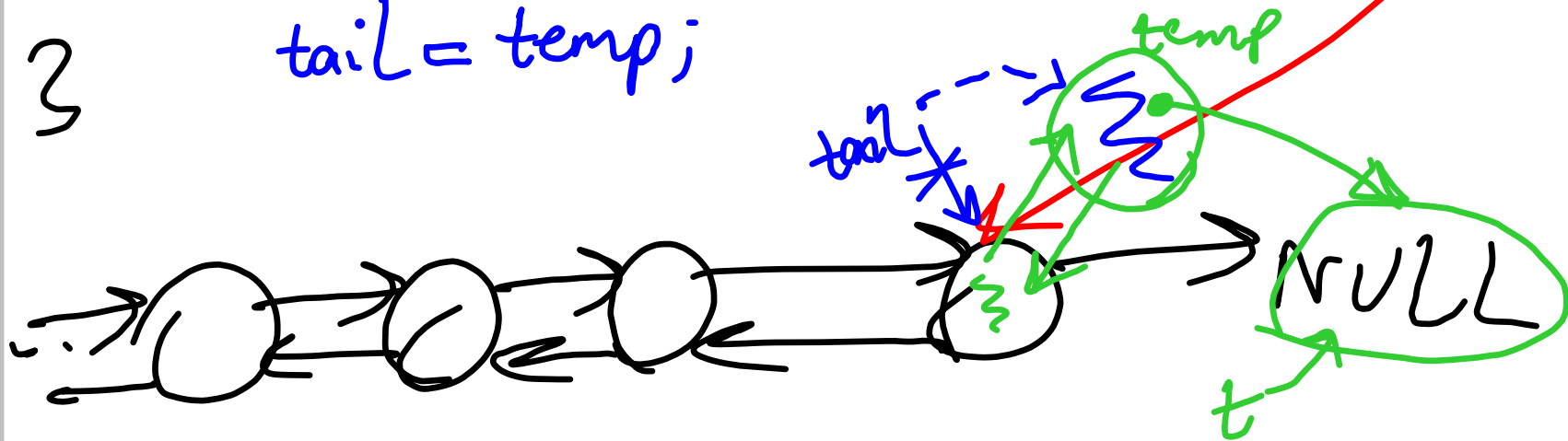


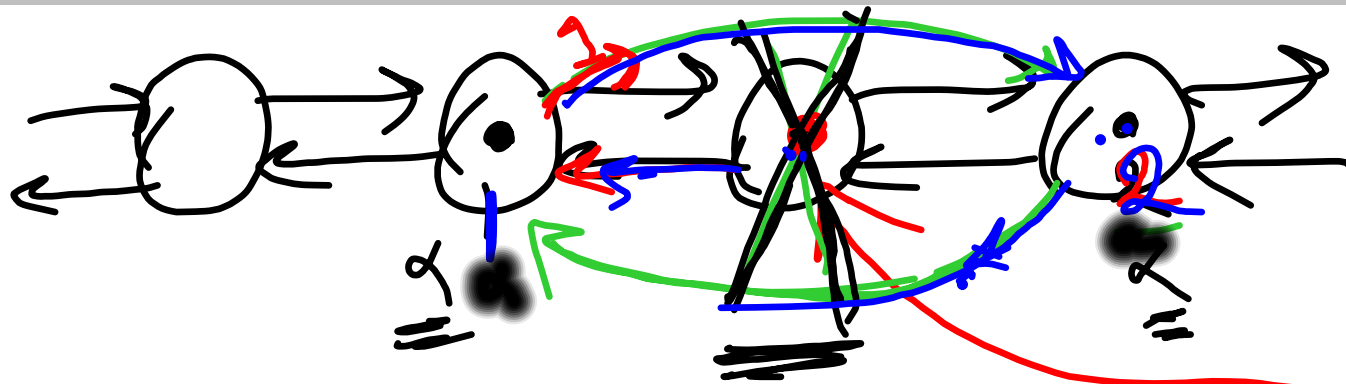
```
void List<X>::insertAfter ( NodeList<X> *
```

```

- ListNode<X> * temp = new ListNode<X> (
- ListNode<X> * t = node -> next;
- node -> next = temp;
- temp -> prev = node
- temp -> next = t;
  if (t != NULL)
    t -> prev = temp;
  else
    tail = temp;
}

```





List <X> !!

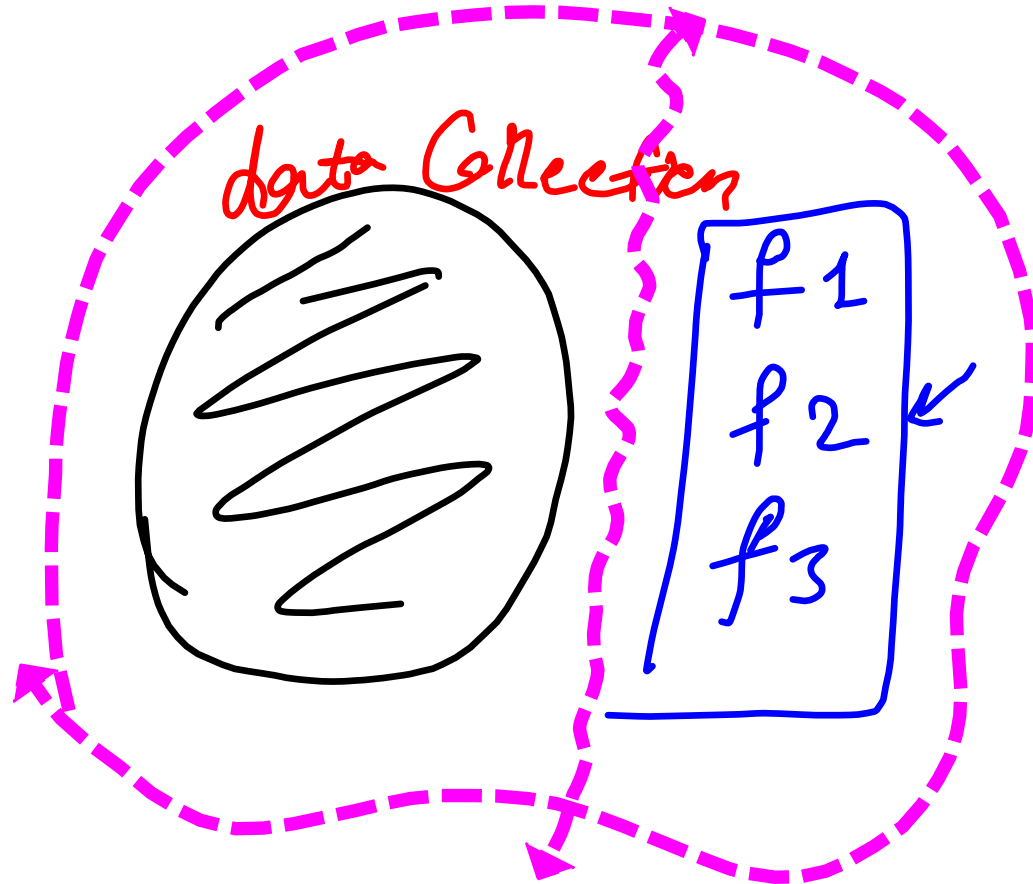
```
void remove ( NodeList<X> * node ) {
```

```

    node -> prev -> next = node -> next;
    node -> next -> prev = node -> prev;
    delete node;
}

```

ADT Abstract Data Type



Array <int>

1, 2, 19e, 2e1, -5

