



# Foundational Data Structures

# Arrays

- Fixed, Contiguous Blocks of Memory of the same type
- Pros
  - Random Access
  - Great for Data Locality
- Cons
  - Cannot Resize
  - Not great if the size is not known or fixed

## Array of Size 10 In Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| a[]        | 36       | 28           |
| ...        | ...      | ...          |
| a[0]       | 256      | 36           |
| a[1]       |          | 42           |
| a[2]       |          | 48           |
| ...        |          |              |
| a[9]       | NULL     | 90           |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |

# Growable Arrays

- Array Lists

  - Strings

- Growing

  - Create a new array with a larger size
  - Transfer all the data from the original array to the new array
  - Remove the original Array

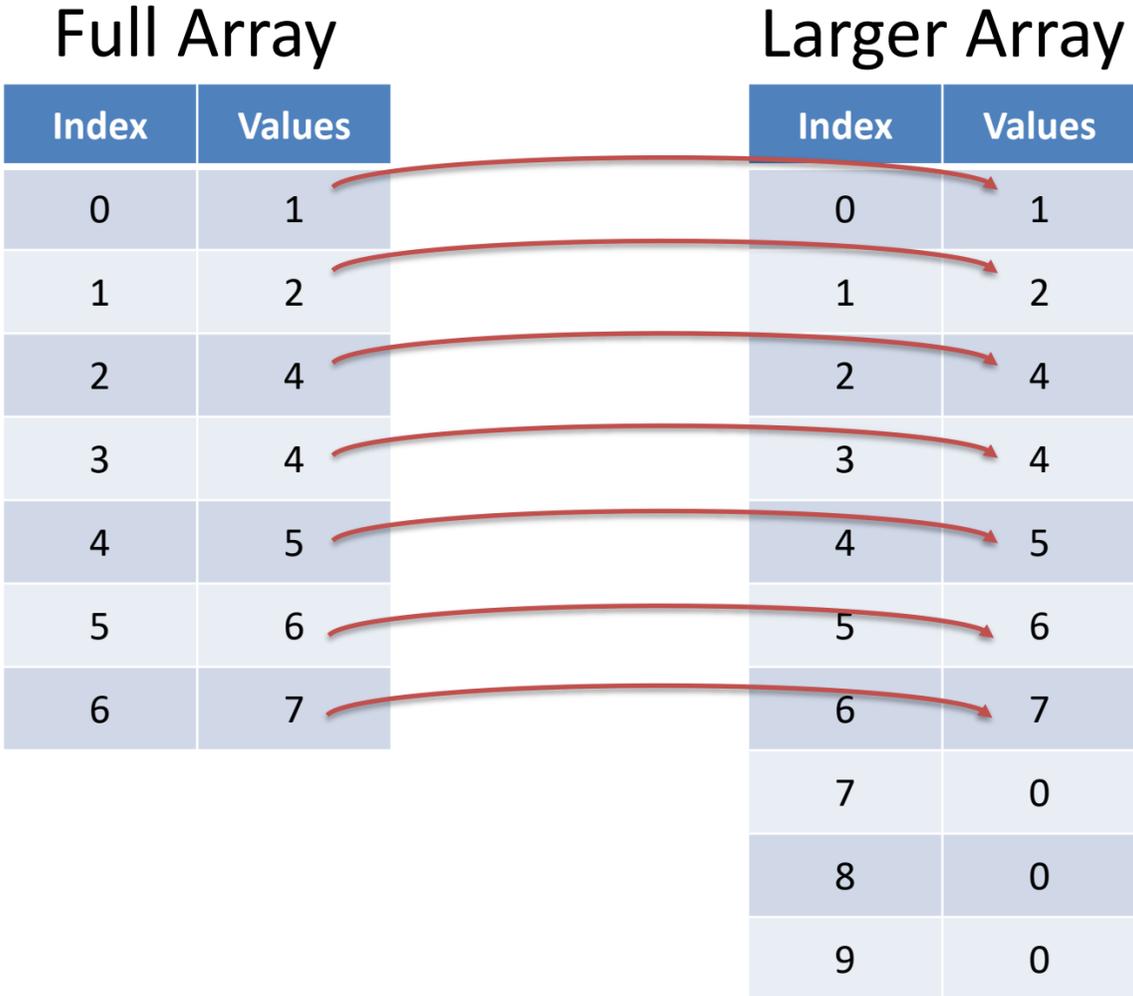
- Pros

  - Semi-Random Access
  - Growable Structure

- Cons

  - Lots of Overhead
  - Does not Shrink
  - Not Great for Large Amounts of Data
  - Not Great Performance

## Growing Concept



IS THERE A BETTER WAY!?



# Linking Structures

Part 01

# Linking Structures

- Groups Together

- Data
- Link(s) / Reference(s) / Pointer(s)
- “Node”

- Pros

- Growable
- Shrinkable

- Cons

- No Random Access
- Not Great for Data Locality

## Node



## List of Nodes



# Linking Concepts

# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

Node



# Linked Lists

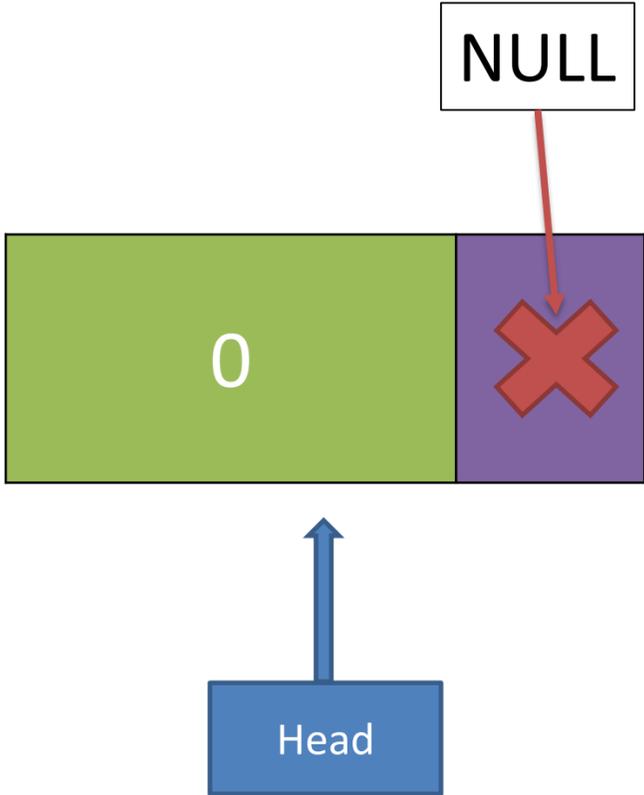
```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
public static void main(String[] args)
{
    //Creating a Node
    Node head = new Node(0,null);
    //Adding another Node after head
    head.link = new Node(1,null);
    //Adding another Node after that
    head.link.link = new Node(2,null);
}
```

# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

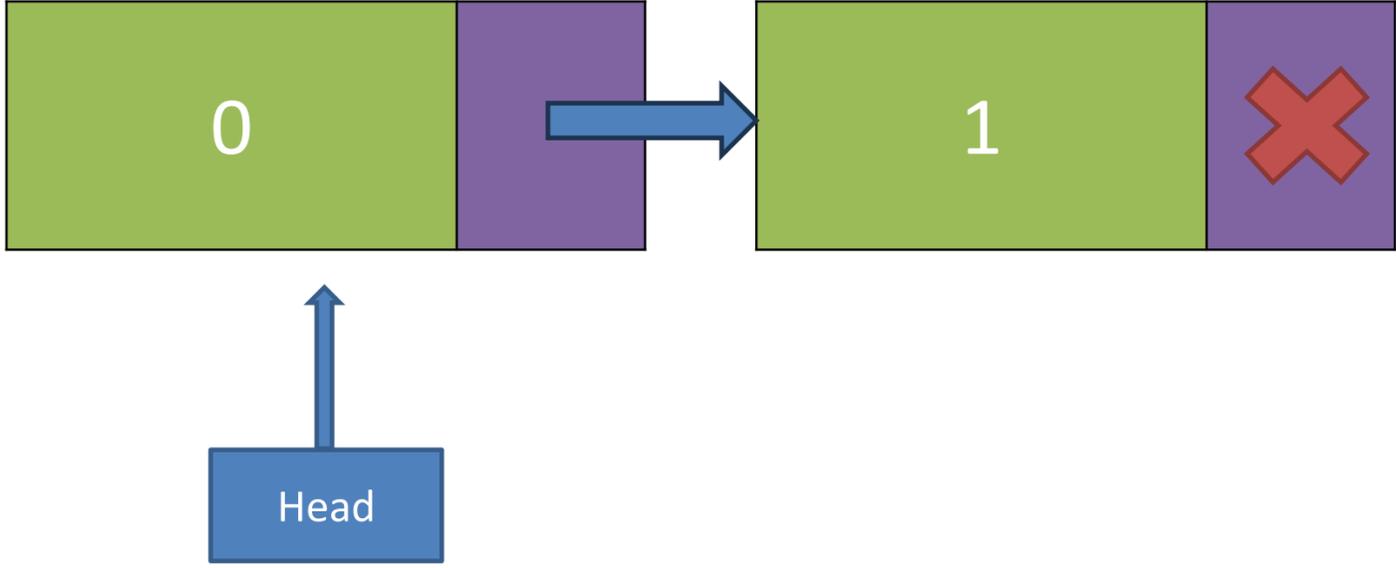
```
public static void main(String[] args)
{
    //Creating a Node
    Node head = new Node(0,null);
    //Adding another Node after head
    head.link = new Node(1,null);
    //Adding another Node after that
    head.link.link = new Node(2,null);
}
```



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
public static void main(String[] args)
{
    //Creating a Node
    Node head = new Node(0,null);
    //Adding another Node after head
    head.link = new Node(1,null);
    //Adding another Node after that
    head.link.link = new Node(2,null);
}
```



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

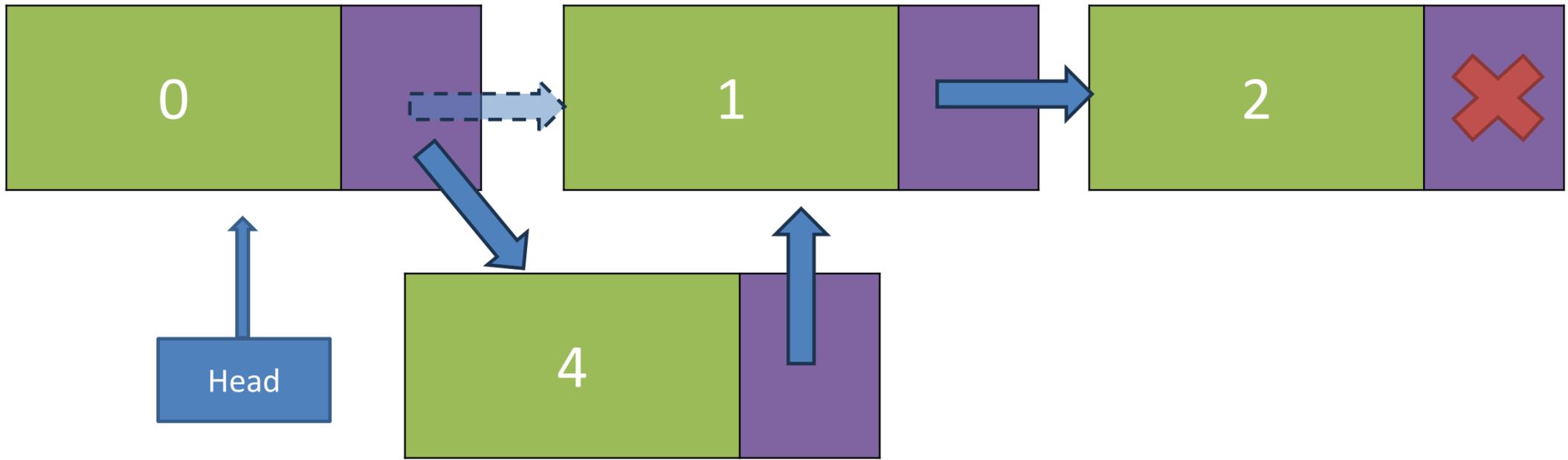
```
public static void main(String[] args)
{
    //Creating a Node
    Node head = new Node(0,null);
    //Adding another Node after head
    head.link = new Node(1,null);
    //Adding another Node after that
    head.link.link = new Node(2,null);
}
```



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

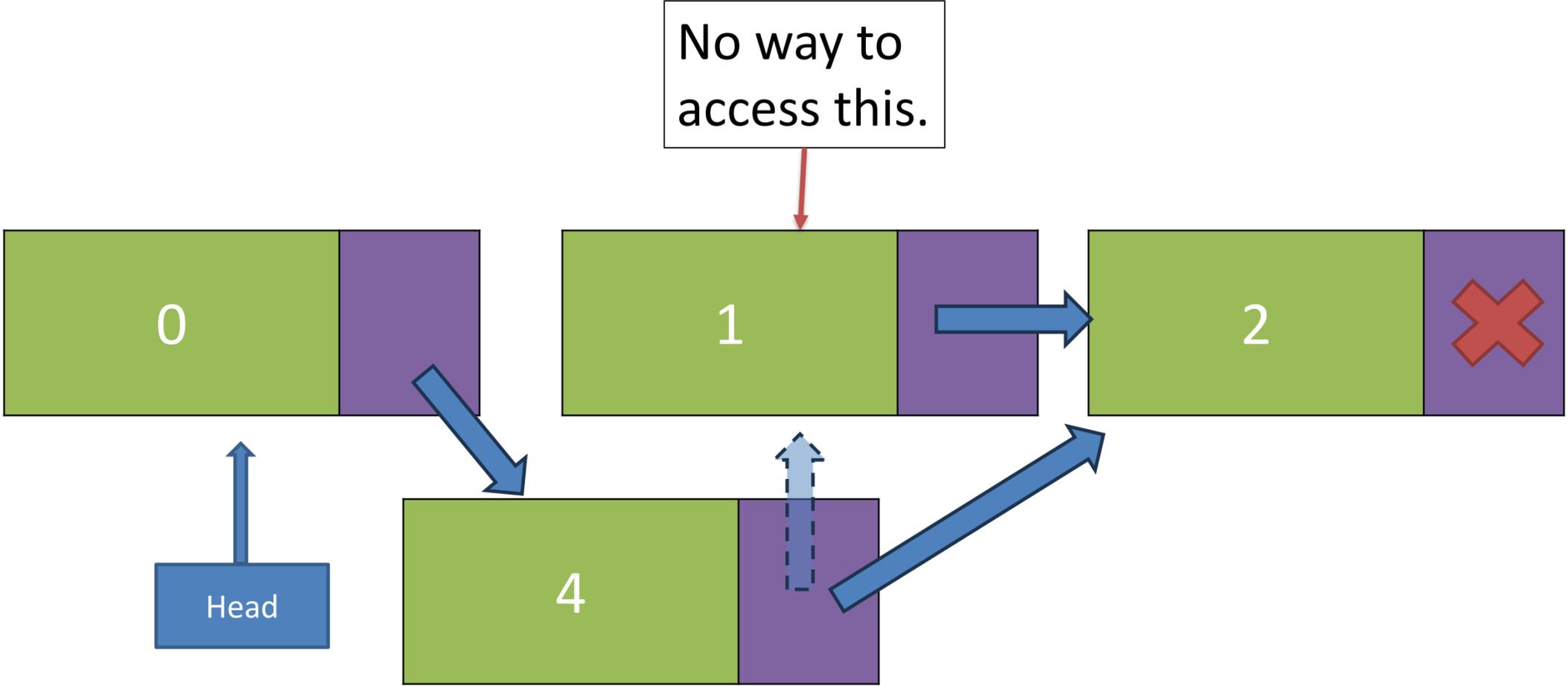
```
public static void main(String[] args)
{
    //Creating a Node
    Node head = new Node(0,null);
    //Adding another Node after head
    head.link = new Node(1,null);
    //Adding another Node after that
    head.link.link = new Node(2,null);
    //Inserting a Node after Head
    head.link = new Node(4,head.link);
}
```



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}

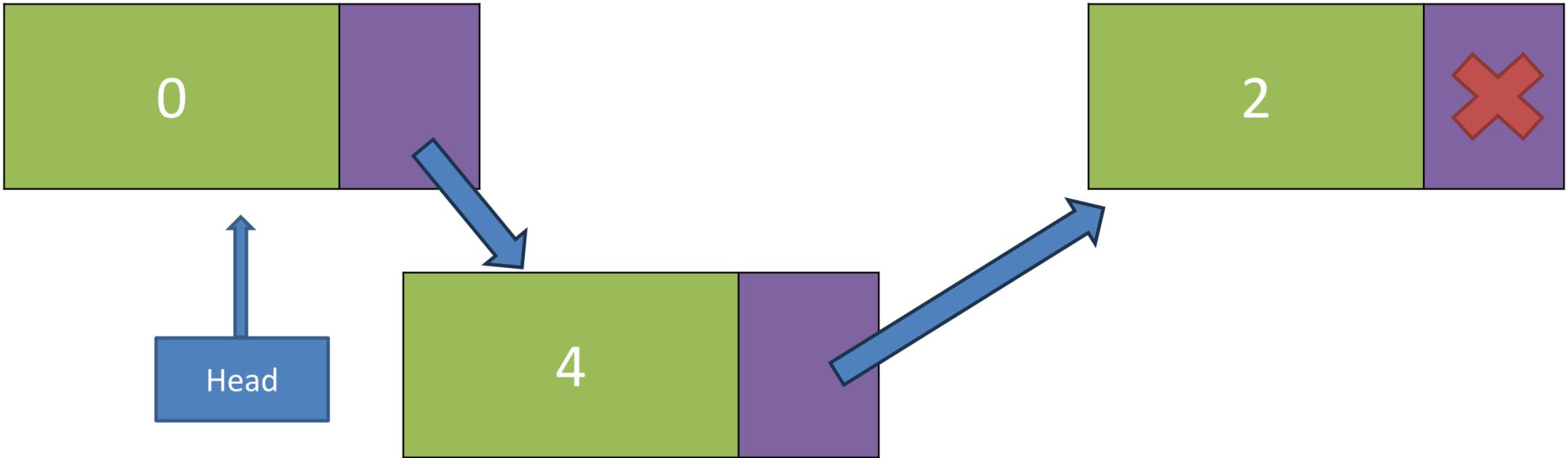
public static void main(String[] args)
{
    //Creating a Node
    Node head = new Node(0,null);
    //Adding another Node after head
    head.link = new Node(1,null);
    //Adding another Node after that
    head.link.link = new Node(2,null);
    //Inserting a Node after Head
    head.link = new Node(4,head.link);
    //Removing the Third Node
    head.link.link = head.link.link.link;
}
```



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}

public static void main(String[] args)
{
    //Creating a Node
    Node head = new Node(0,null);
    //Adding another Node after head
    head.link = new Node(1,null);
    //Adding another Node after that
    head.link.link = new Node(2,null);
    //Inserting a Node after Head
    head.link = new Node(4,head.link);
    //Removing the Third Node
    head.link.link = head.link.link.link;
}
```



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

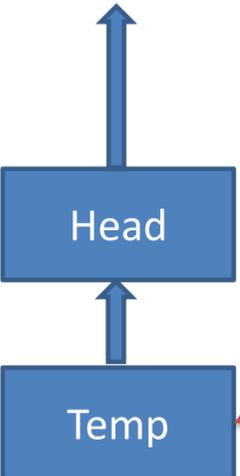
```
Node temp = head;
while(temp != null)
{
    System.out.println(temp.data);
    temp = temp.link;
}
```



# Linked Lists

```
public class Node  
{  
    public int data;  
    public Node link;  
    public Node(int someData, Node aLink)  
    {  
        data = someData;  
        link = aLink;  
    }  
}
```

```
Node temp = head;  
while(temp != null)  
{  
    System.out.println(temp.data);  
    temp = temp.link;  
}
```

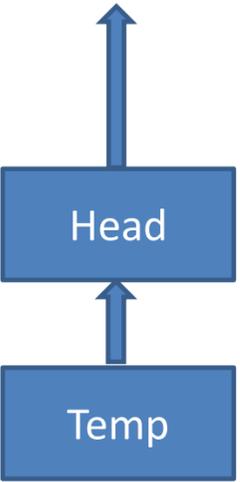


Did not make a *clone*.  
This references the same  
object as "head".

# Linked Lists

```
public class Node  
{  
    public int data;  
    public Node link;  
    public Node(int someData, Node aLink)  
    {  
        data = someData;  
        link = aLink;  
    }  
}
```

```
Node temp = head;  
while(temp != null) TRUE!  
{  
    System.out.println(temp.data);  
    temp = temp.link;  
}
```

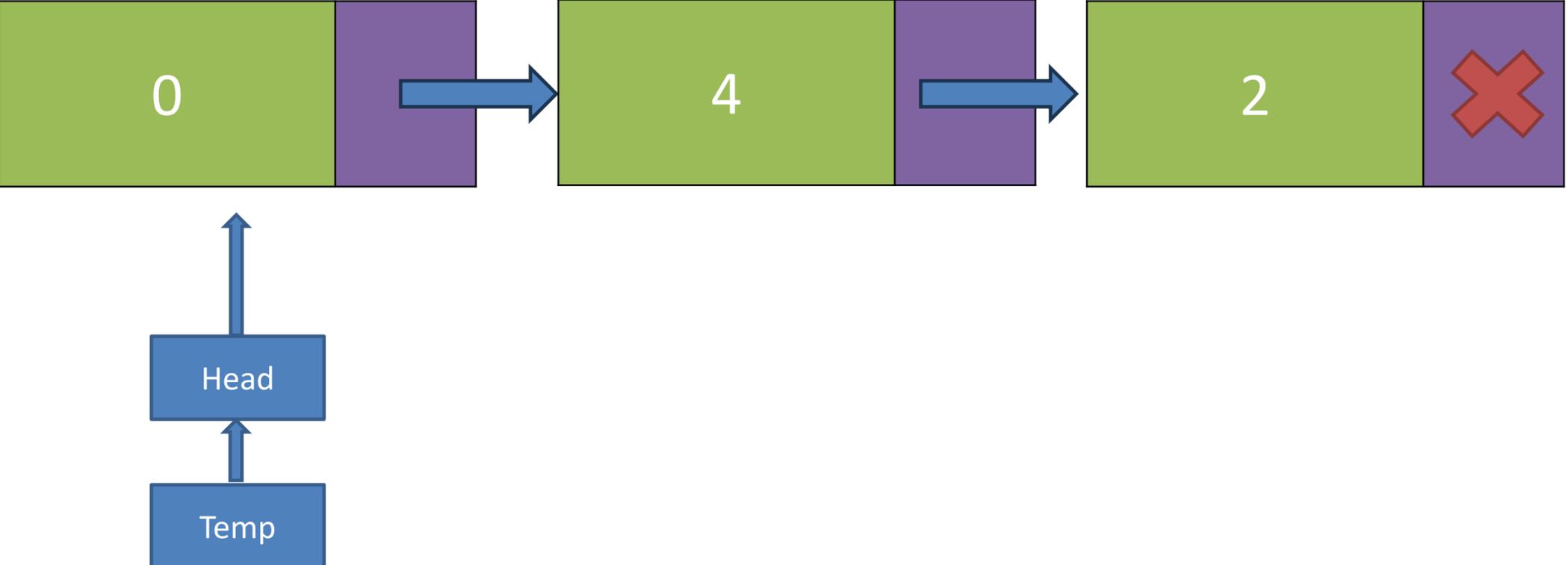


# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
Node temp = head;
while(temp != null)
{
    System.out.println(temp.data);
    temp = temp.link;
}
```

Console  
0

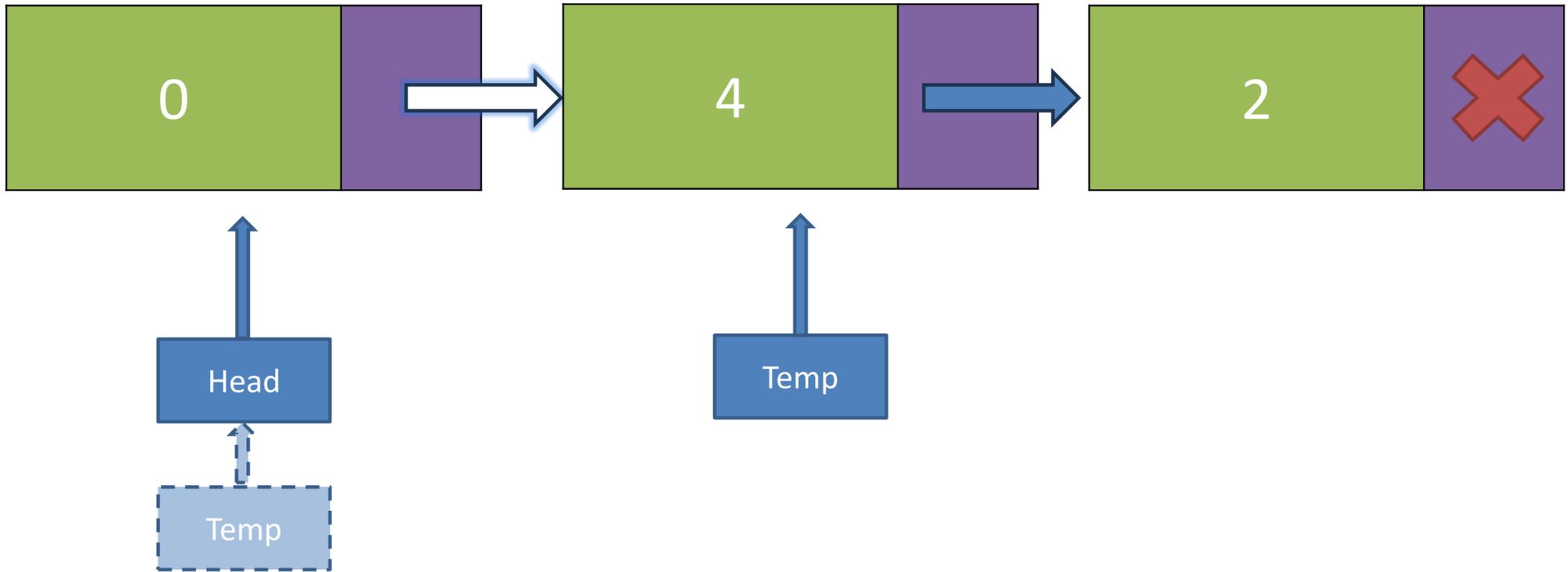


# Linked Lists

```
public class Node  
{  
    public int data;  
    public Node link;  
    public Node(int someData, Node aLink)  
    {  
        data = someData;  
        link = aLink;  
    }  
}
```

```
Node temp = head;  
while(temp != null)  
{  
    System.out.println(temp.data);  
    temp = temp.link;  
}
```

Console  
0

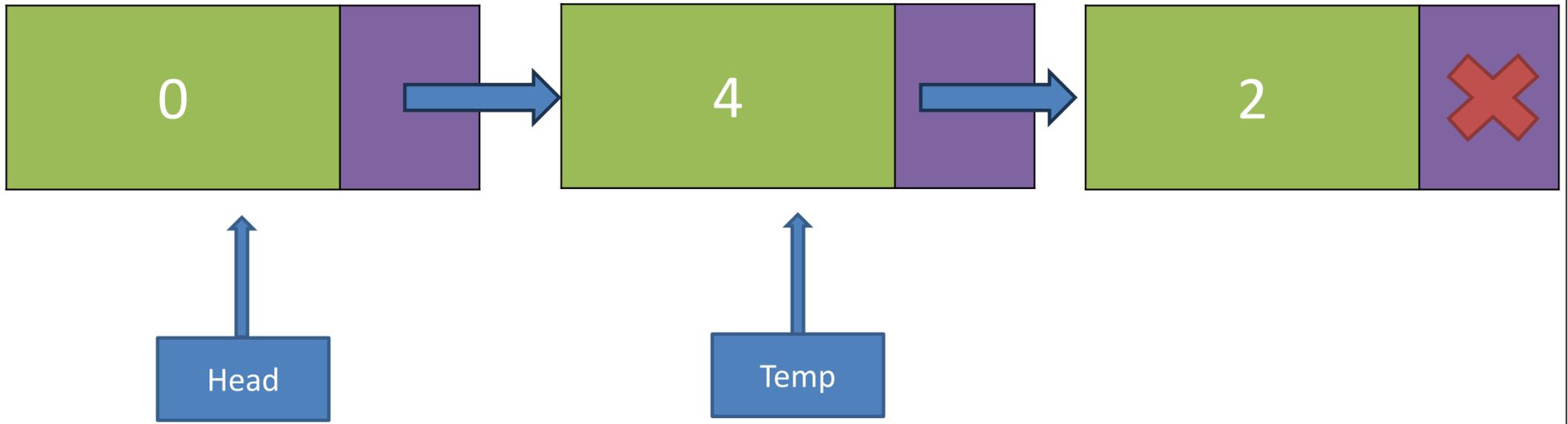


# Linked Lists

```
public class Node  
{  
    public int data;  
    public Node link;  
    public Node(int someData, Node aLink)  
    {  
        data = someData;  
        link = aLink;  
    }  
}
```

```
Node temp = head;  
while(temp != null) TRUE!  
{  
    System.out.println(temp.data);  
    temp = temp.link;  
}
```

Console  
0

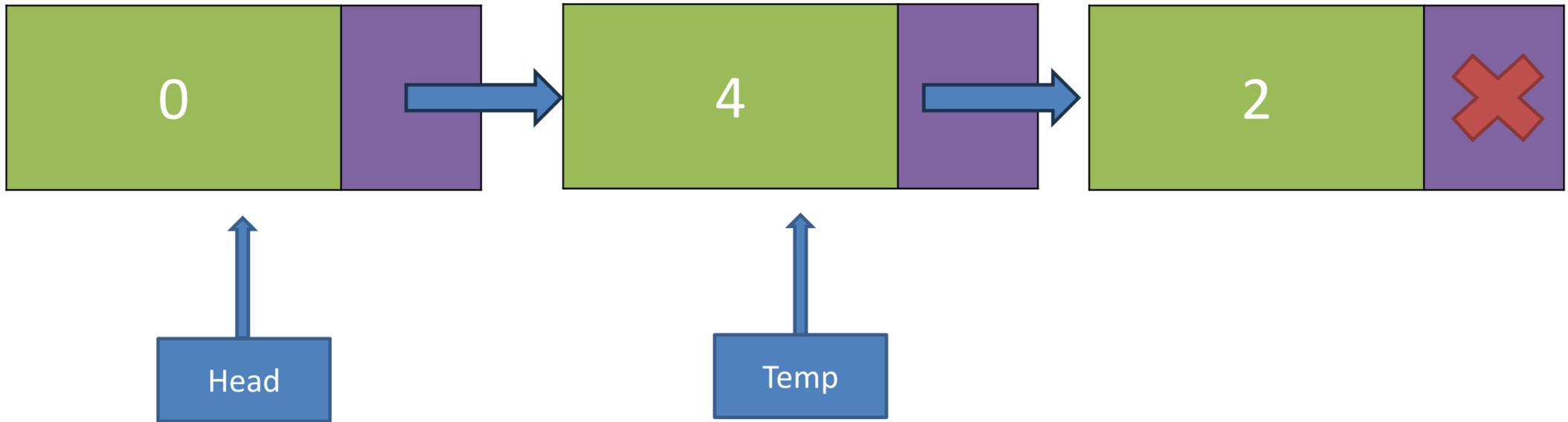


# Linked Lists

```
public class Node  
{  
    public int data;  
    public Node link;  
    public Node(int someData, Node aLink)  
    {  
        data = someData;  
        link = aLink;  
    }  
}
```

```
Node temp = head;  
while(temp != null)  
{  
    System.out.println(temp.data);  
    temp = temp.link;  
}
```

Console  
0  
4



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
Node temp = head;
while(temp != null)
{
    System.out.println(temp.data);
    temp = temp.link;
}
```

Console  
0  
4



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
Node temp = head;
while(temp != null)
{
    System.out.println(temp.data);
    temp = temp.link;
}
```

TRUE!

Console

0  
4



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
Node temp = head;
while(temp != null)
{
    System.out.println(temp.data);
    temp = temp.link;
}
```

Console

```
0
4
2
```



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
Node temp = head;
while(temp != null)
{
    System.out.println(temp.data);
    temp = temp.link;
}
```

Console  
0  
4  
2



# Linked Lists

```
public class Node
{
    public int data;
    public Node link;
    public Node(int someData, Node aLink)
    {
        data = someData;
        link = aLink;
    }
}
```

```
Node temp = head;
while(temp != null) FALSE!
{
    System.out.println(temp.data);
    temp = temp.link;
}
```

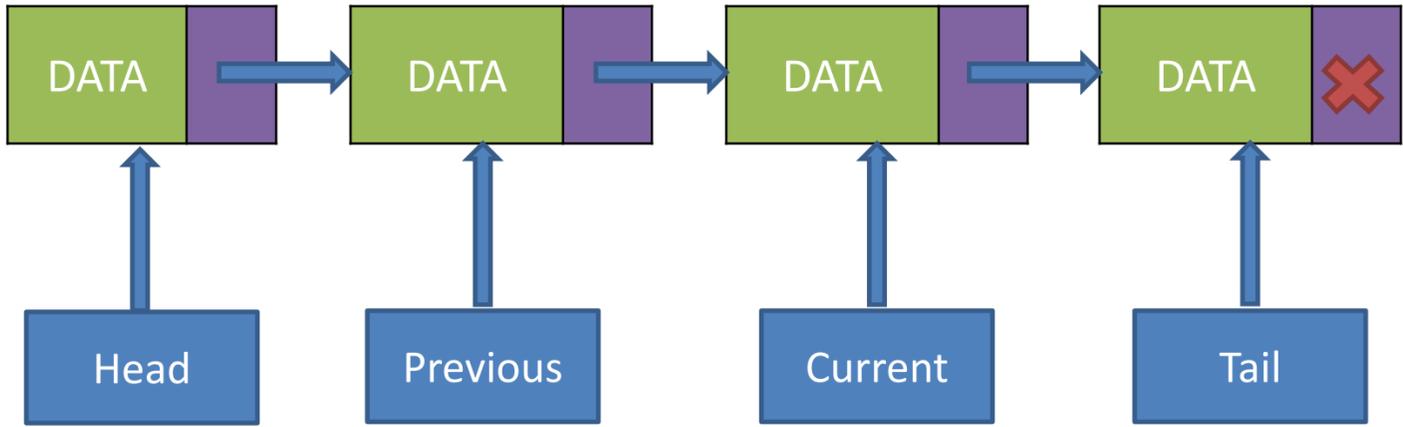
Console  
0  
4  
2



# Linked Lists

- Nodes Contain
  - Data
  - Link
- Special Nodes
  - Head: Always points to the first element of the list
  - Tail: Always points to the last element of the list
  - Current: Movable pointer used to Access and Modify Data in the List
  - Previous: Always stays on node behind Current
- Certain Linked Lists may omit some of these Nodes

## Linked List



# Linked List Set Up

# Internal Classes

- Classes within Classes
- Aids in grouping together like-information that is only used within a class.
- Other Programmers do not need access to these classes.

## Syntax

```
public <<class identifier>>
{
    protected <<internal class identifier>>
    {
        //Body of Internal Class
    }
}
```

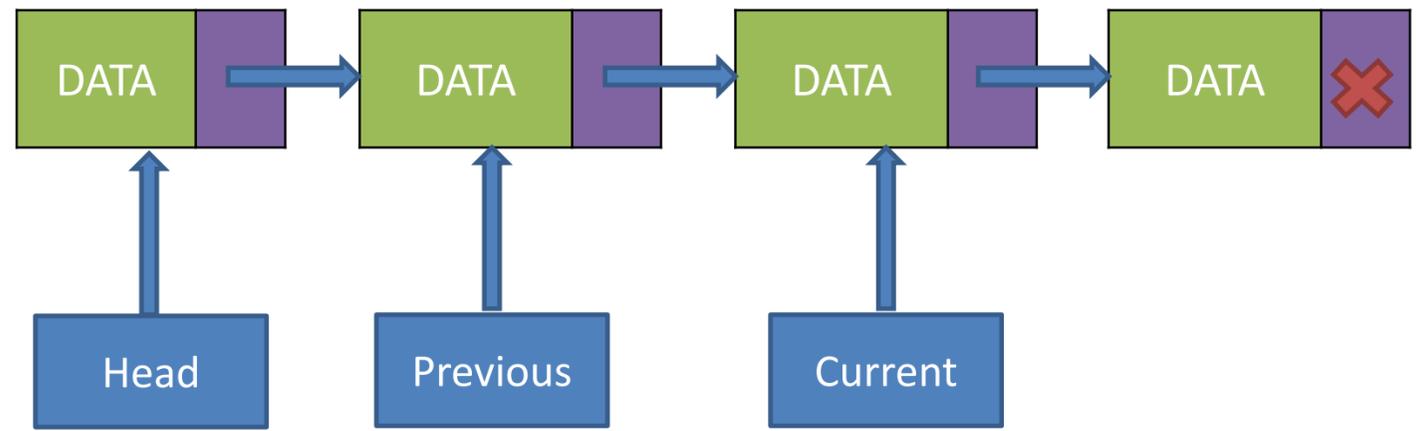
## Example

```
public class IntLL
{
    protected class ListNode
    {
    }
}
```

# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

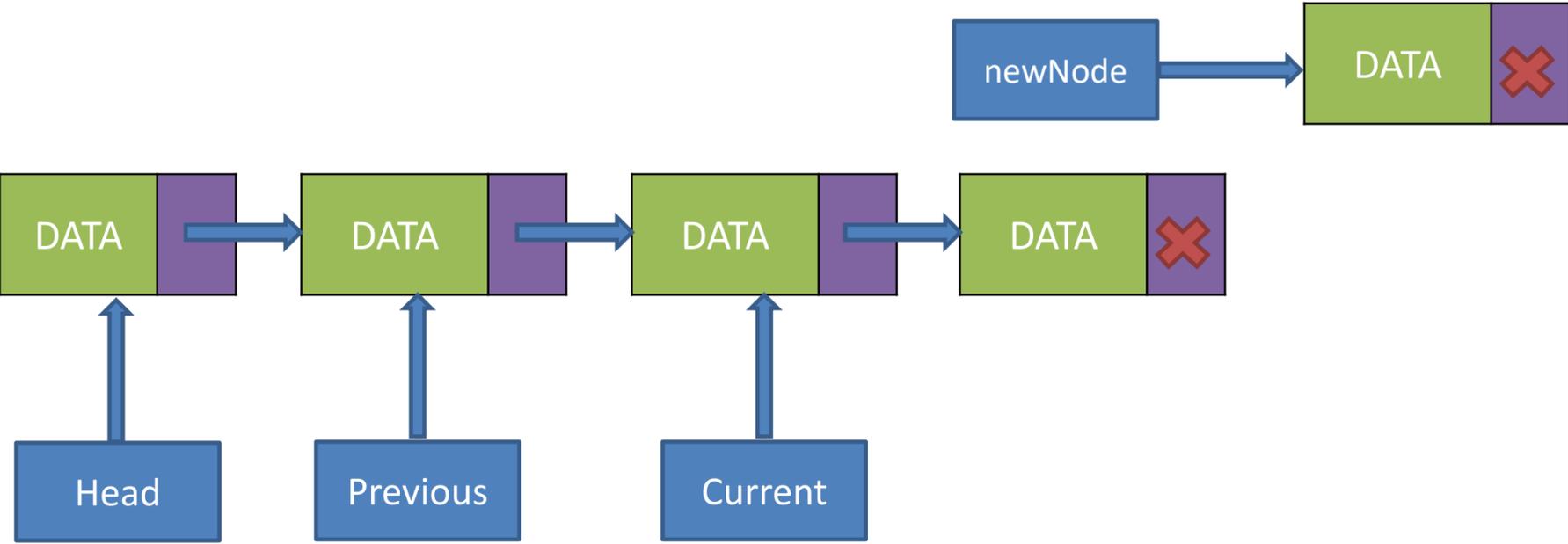
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

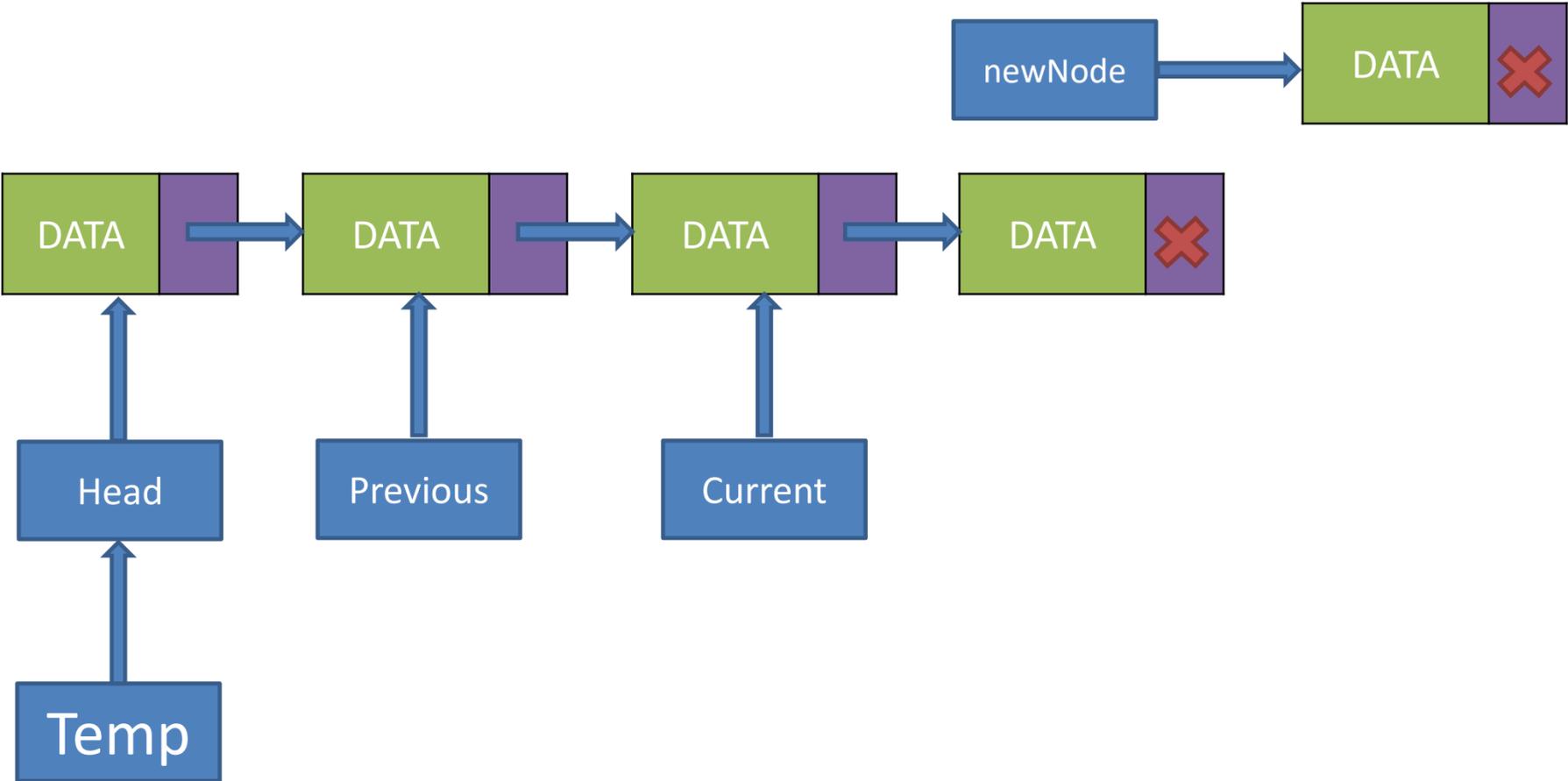
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

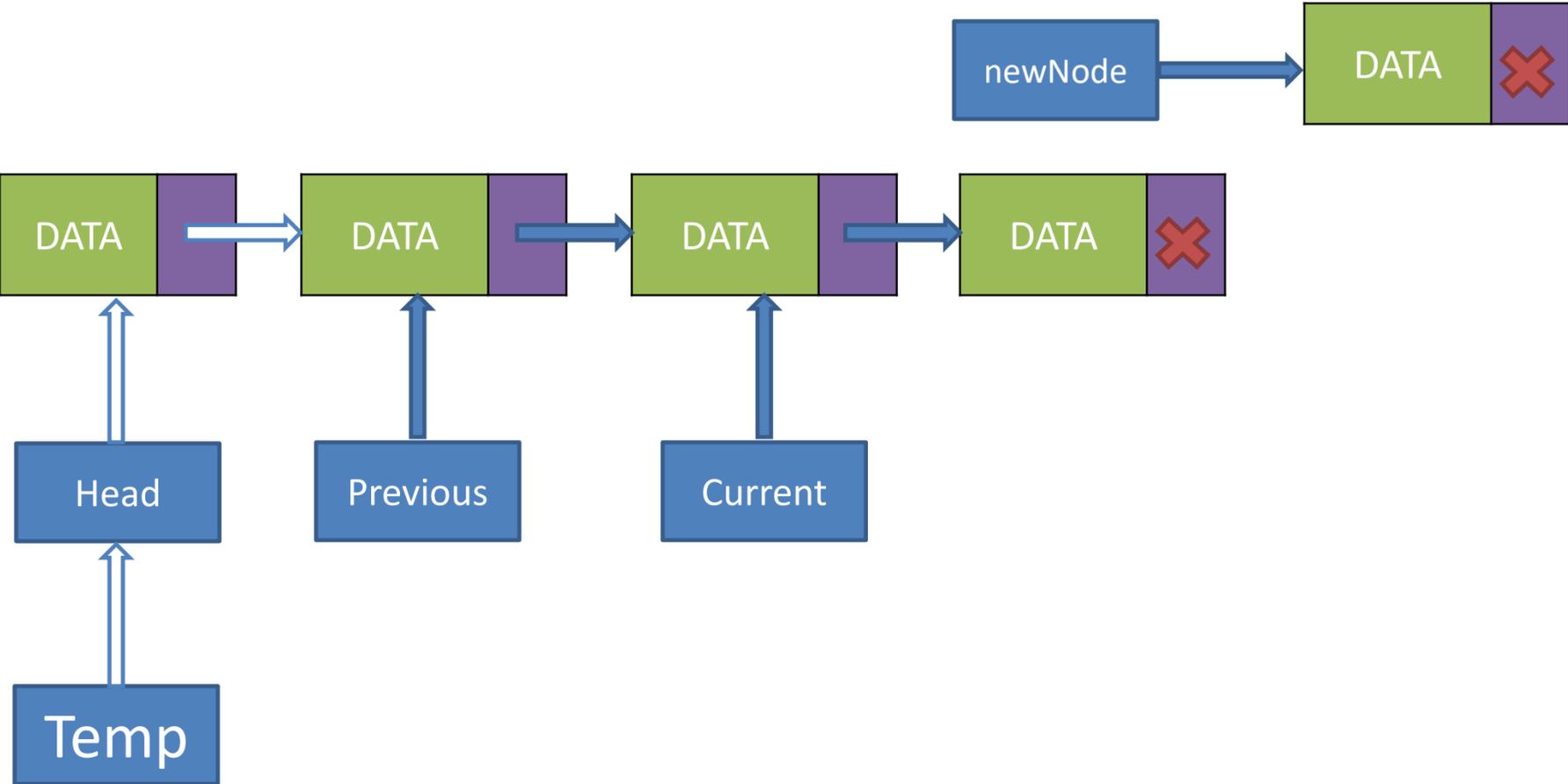
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

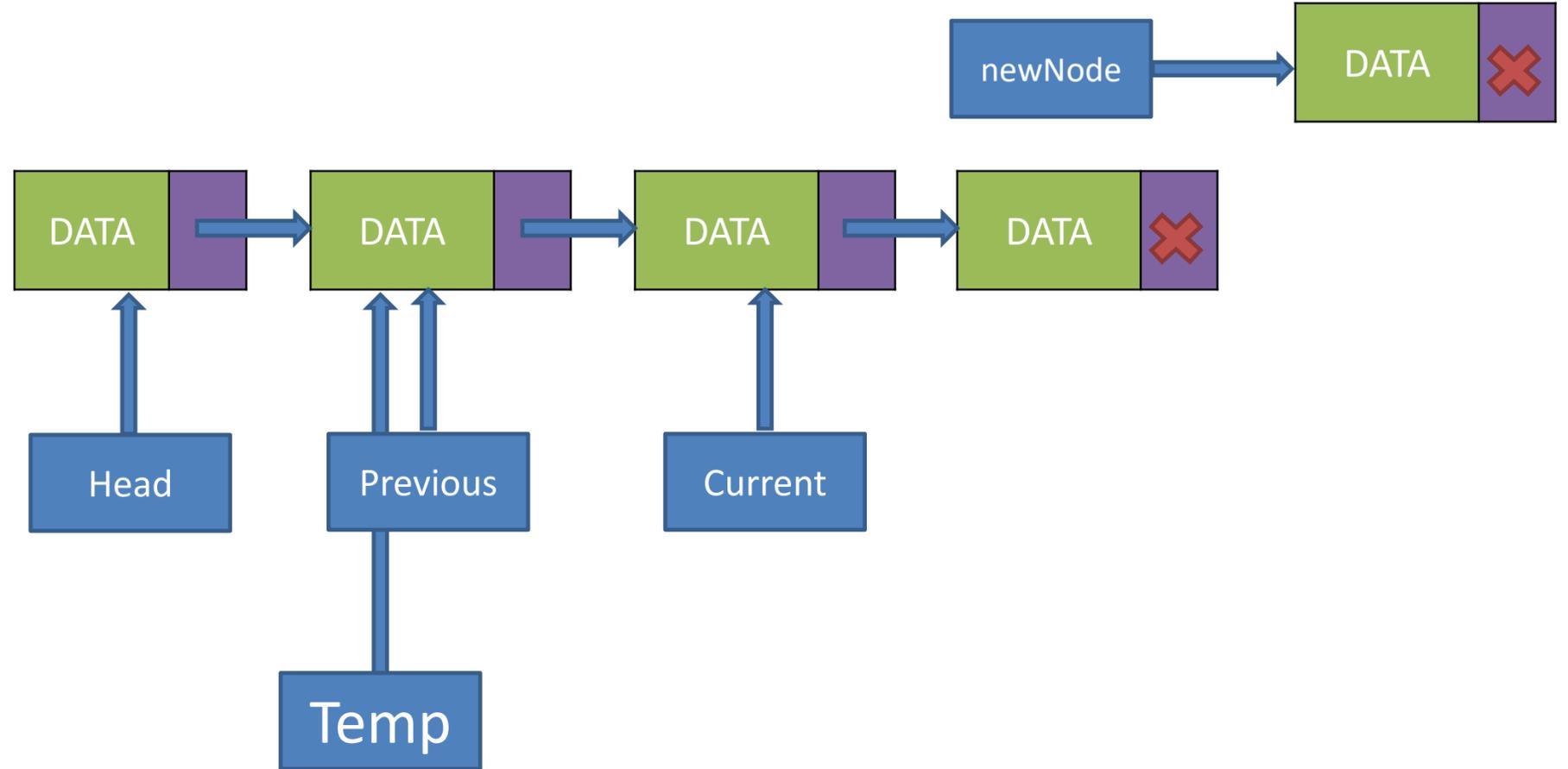
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

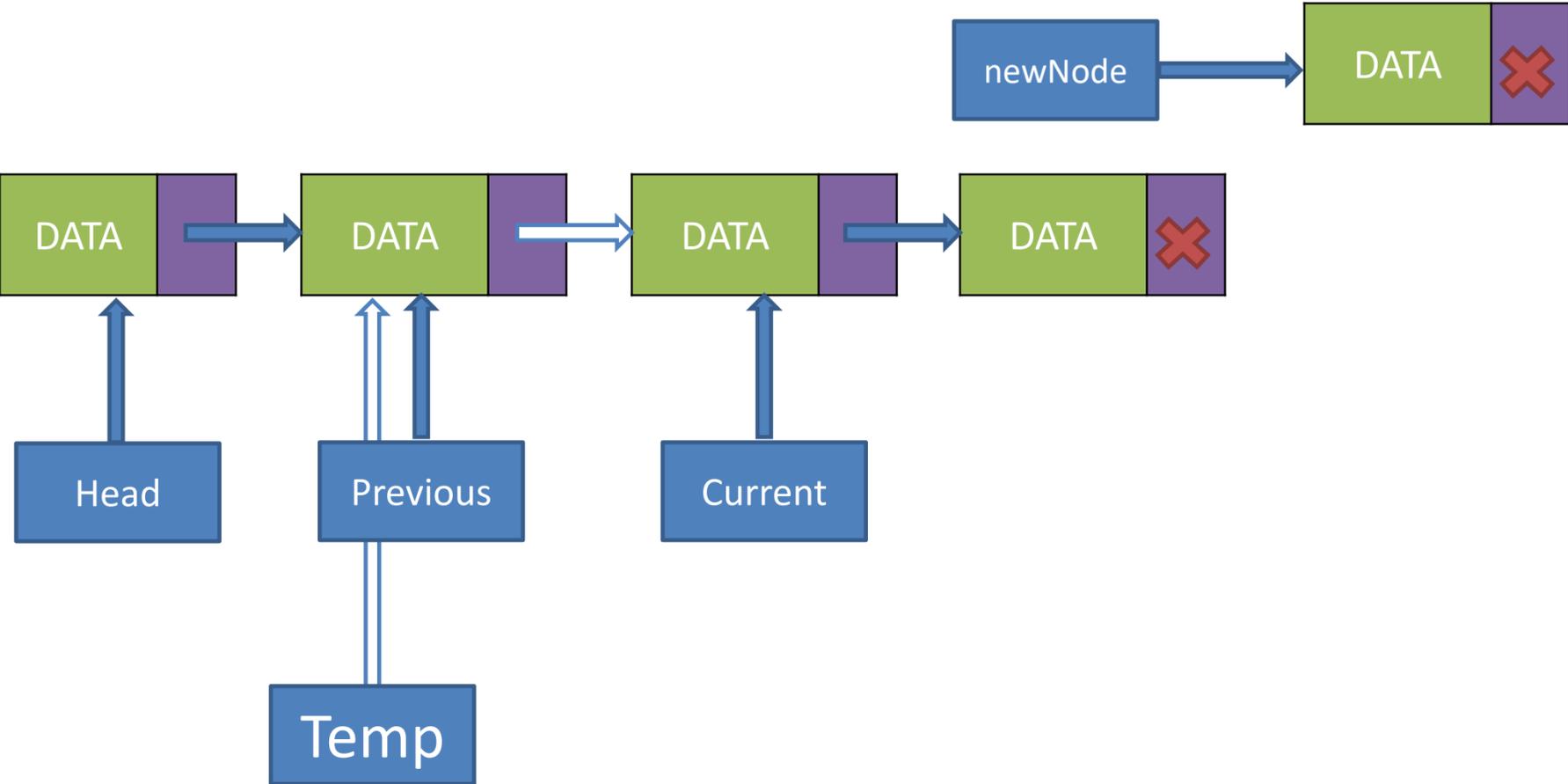
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

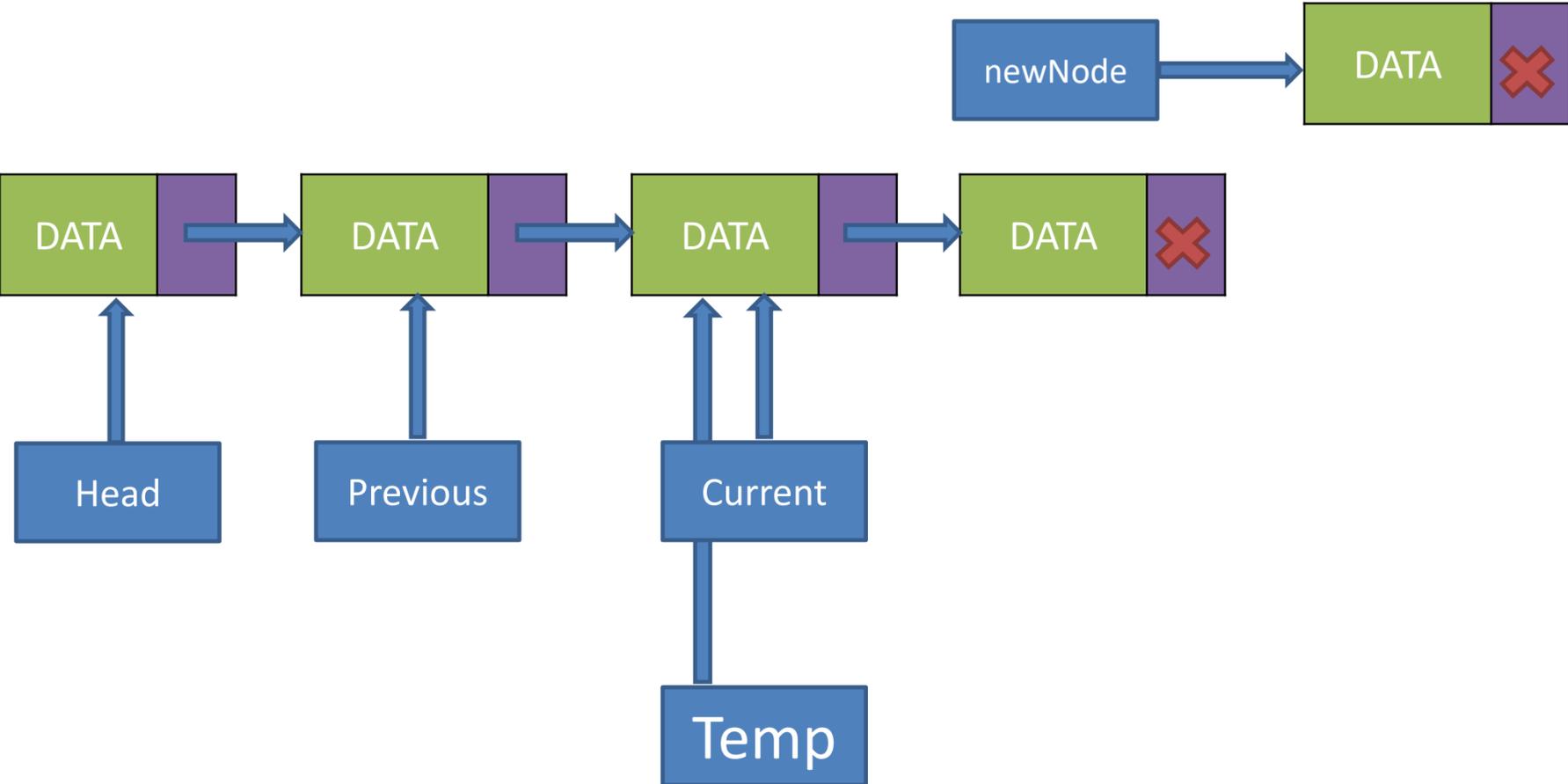
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

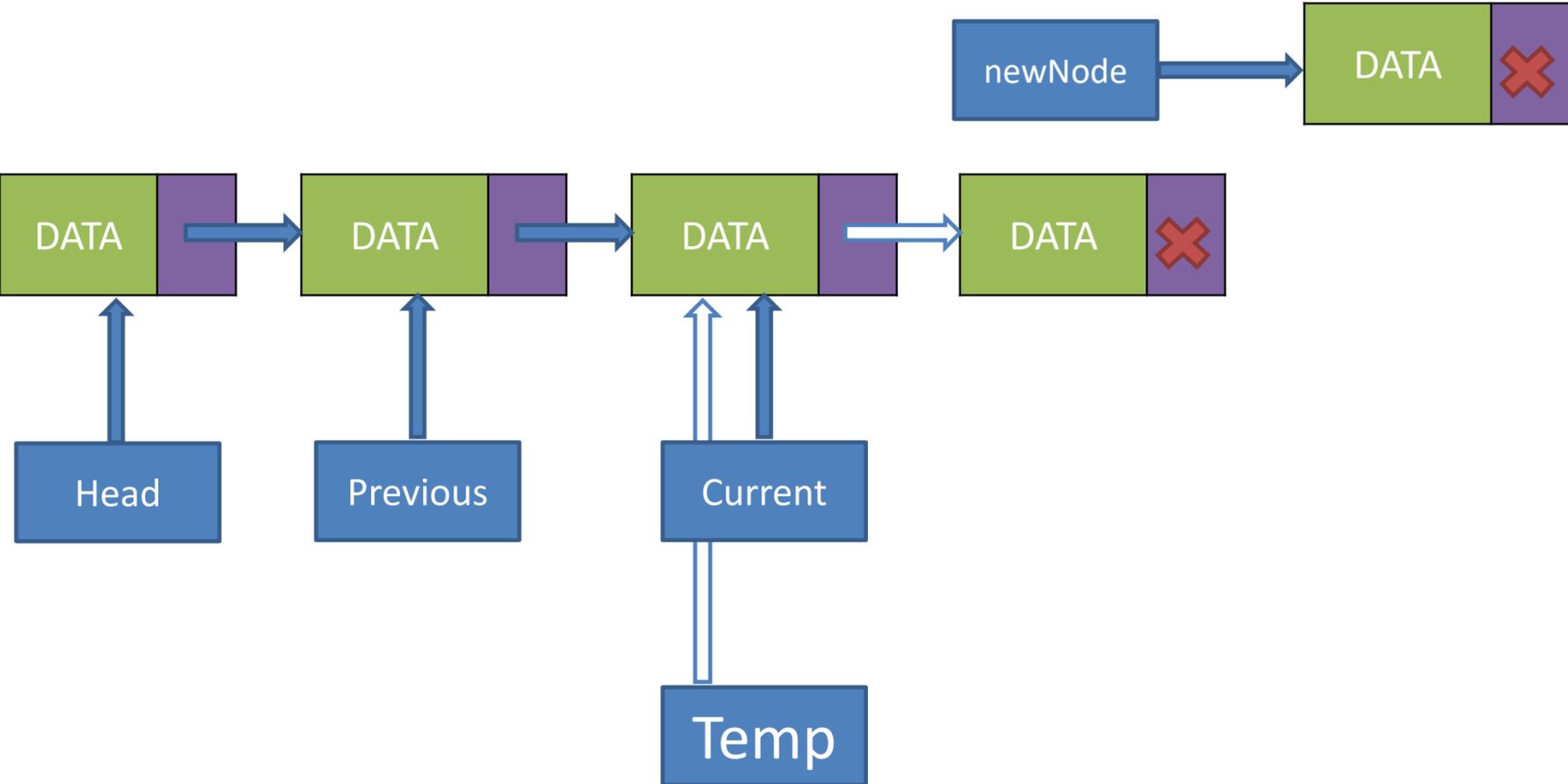
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

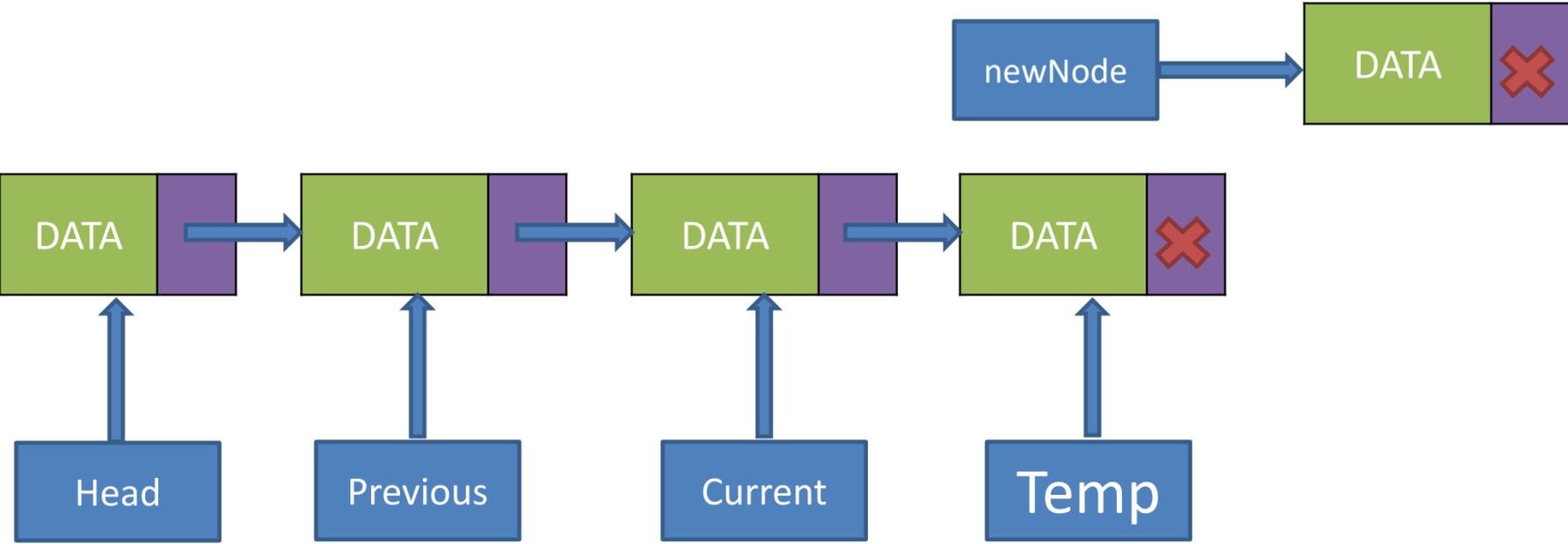
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

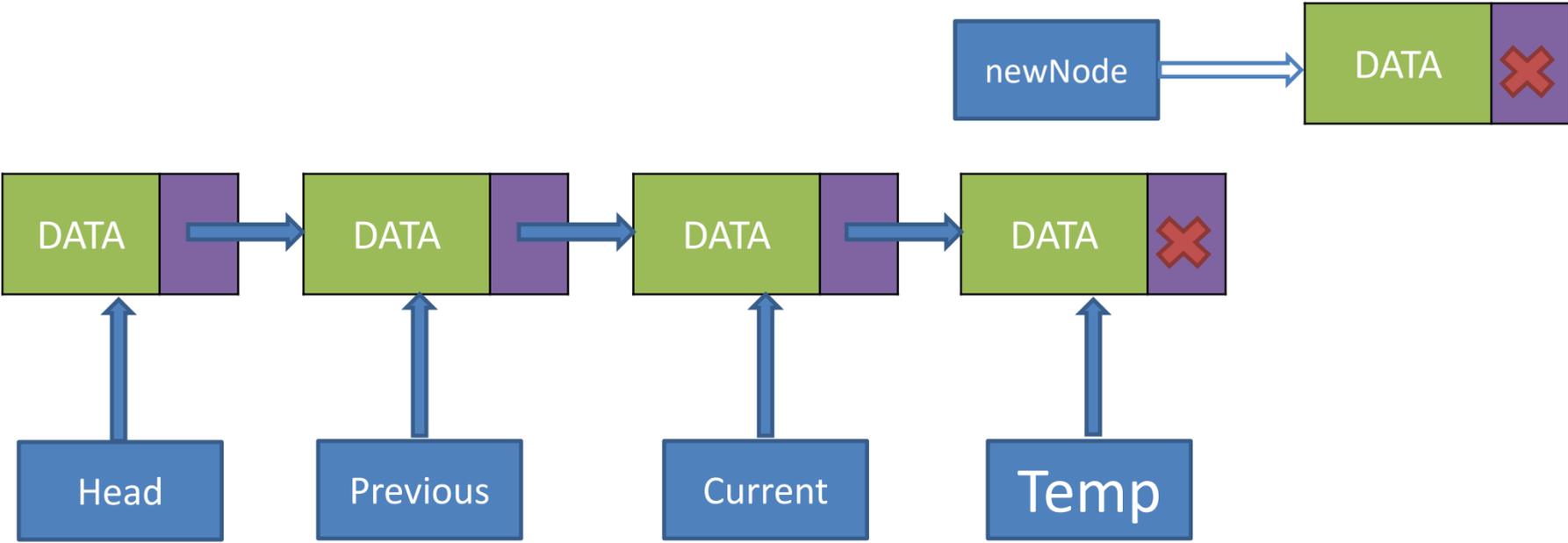
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

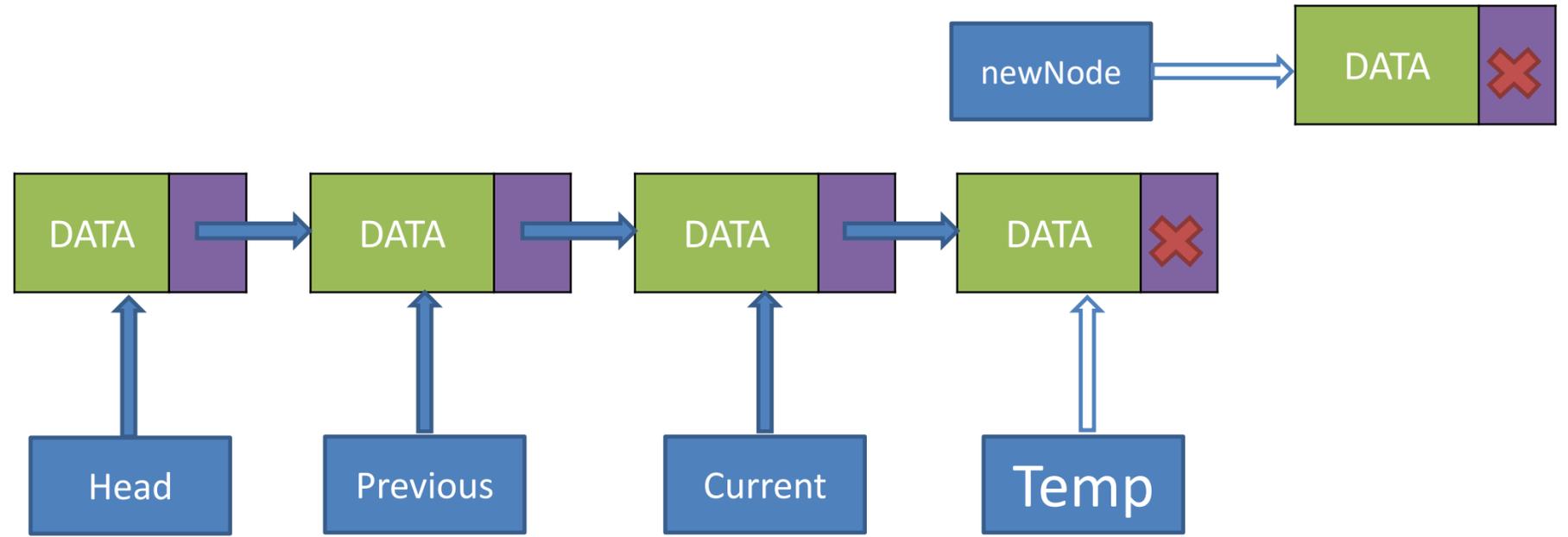
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

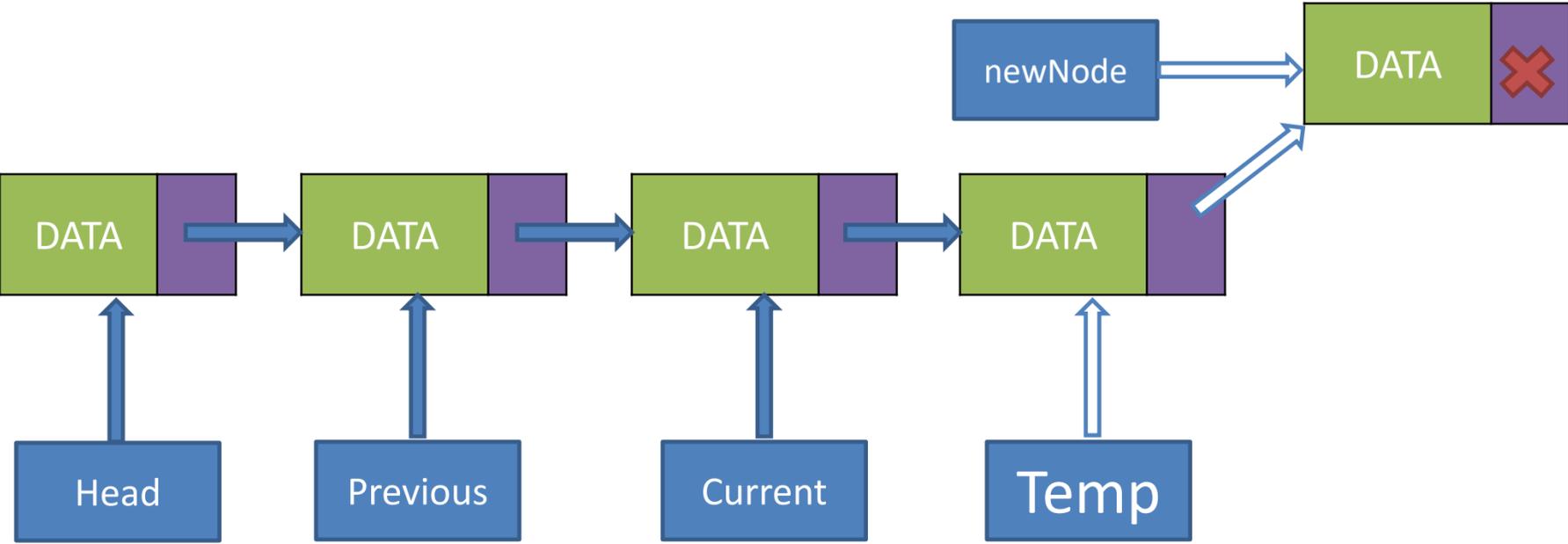
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

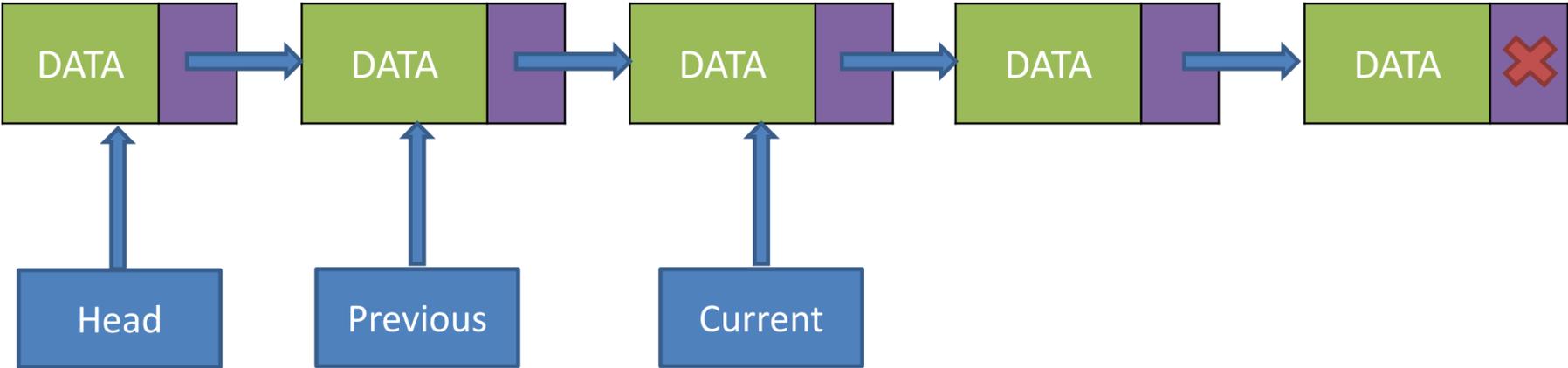
## Concept



# Adding

- Create a new Node with the given Data
- Start from the Head and find the Node with the first Null Link
- Point that Node to the newly Created Node

## Concept



























# Adding in Memory

4

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

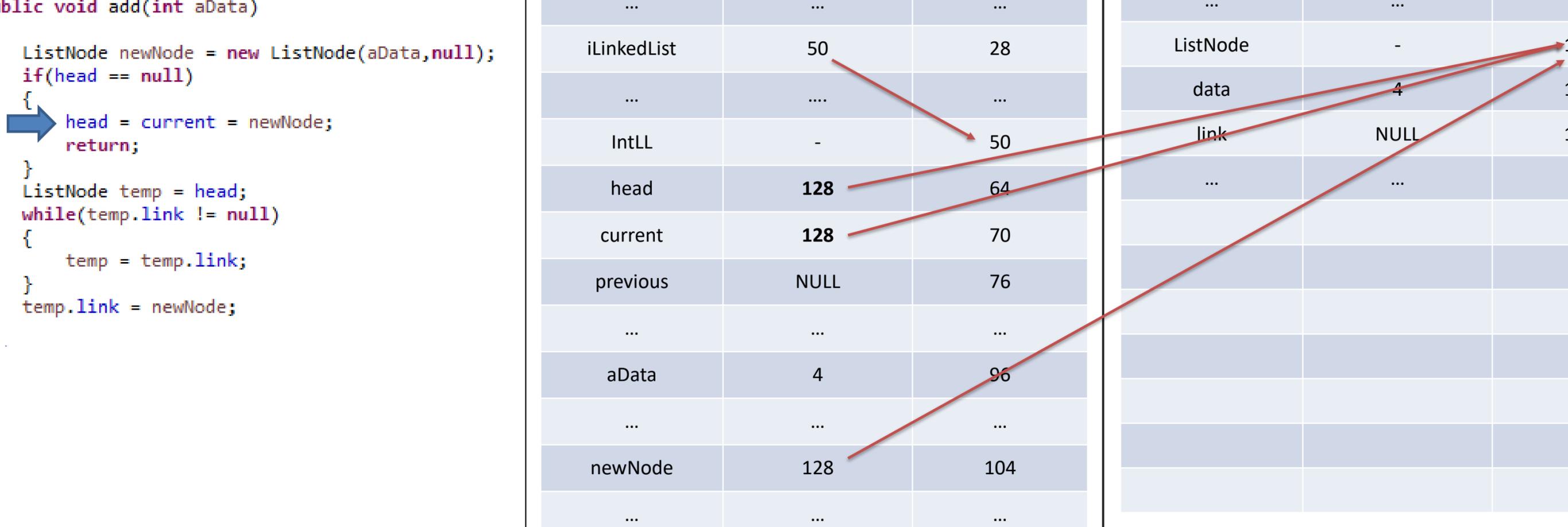
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 4        | 96           |
| ...         | ...      | ...          |
| newNode     | 128      | 104          |
| ...         | ...      | ...          |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |





# Adding in Memory

4

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

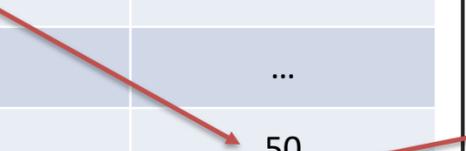
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |



# Adding in Memory

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |

# Adding in Memory

3

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |



# Adding in Memory

3

```

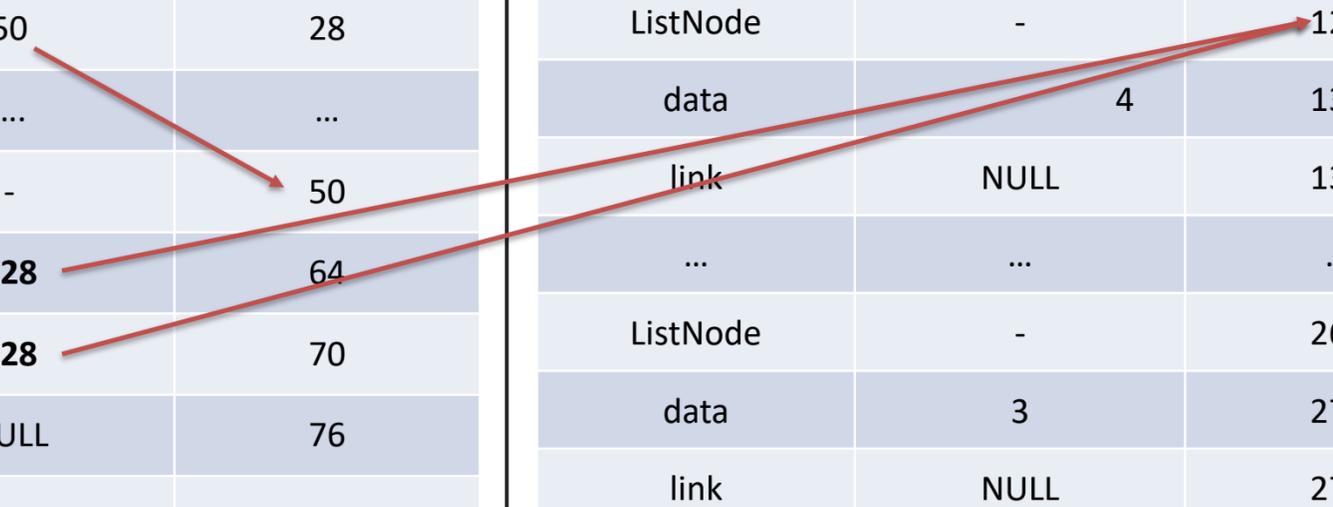
public void add(int aData)
{
    ← ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}
    
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

public void add(int aData)
{
    ← ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

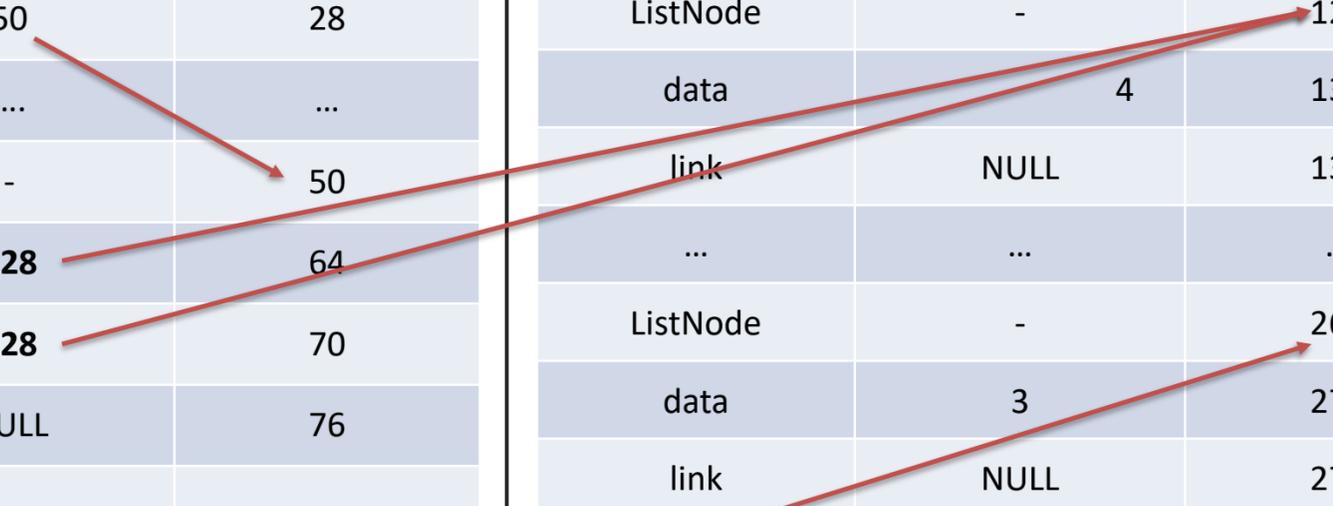
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

public void add(int aData)
{
  ListNode newNode = new ListNode(aData,null);
  if(head == null)
  {
    head = current = newNode;
    return;
  }
  ListNode temp = head;
  while(temp.link != null)
  {
    temp = temp.link;
  }
  temp.link = newNode;
}

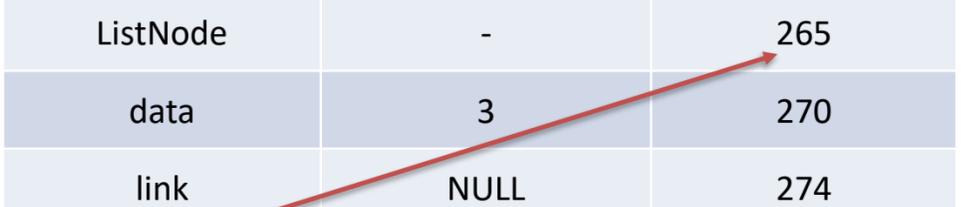
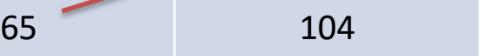
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

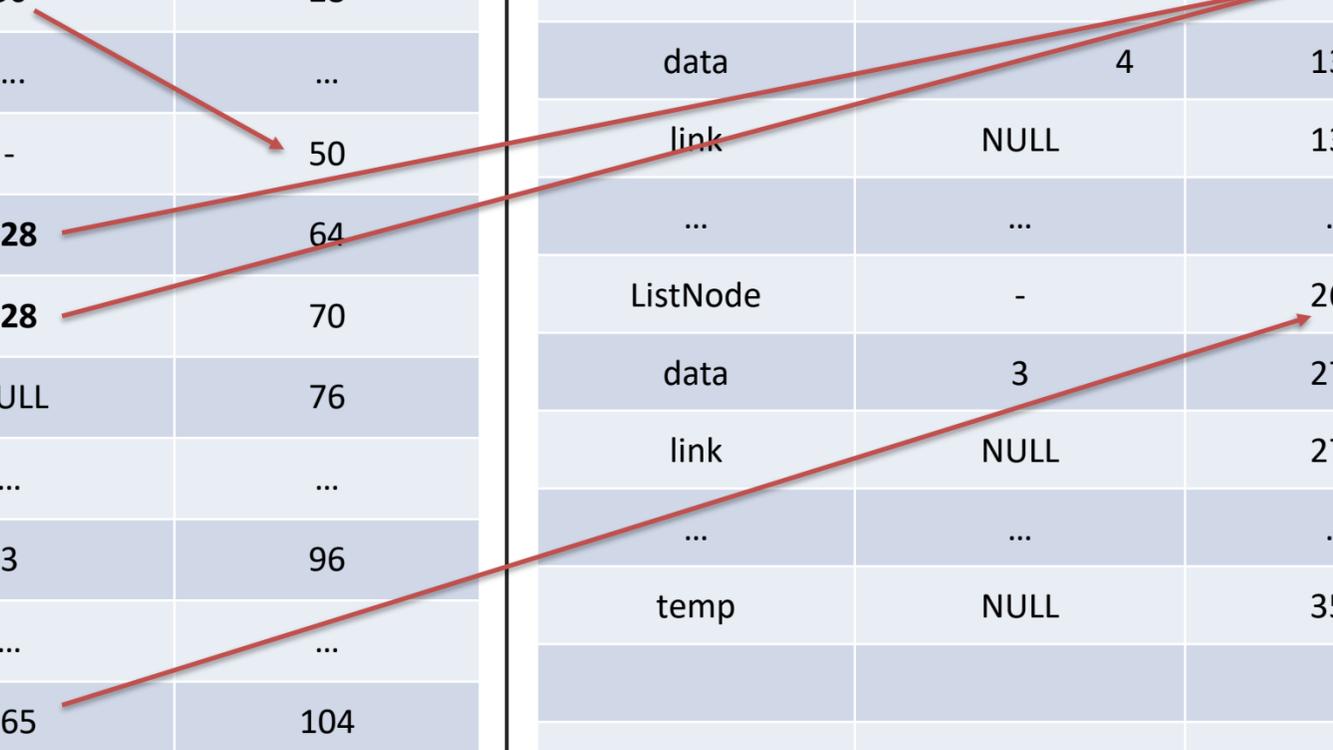
public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    → ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}
    
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| temp       | NULL     | 355          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

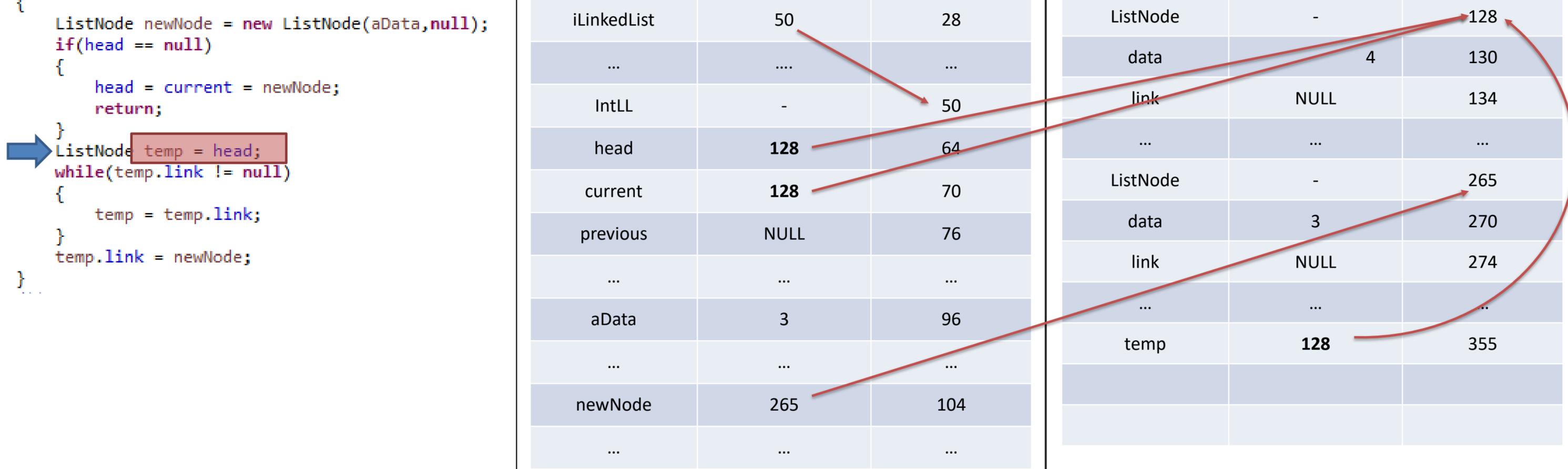
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| temp       | 128      | 355          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

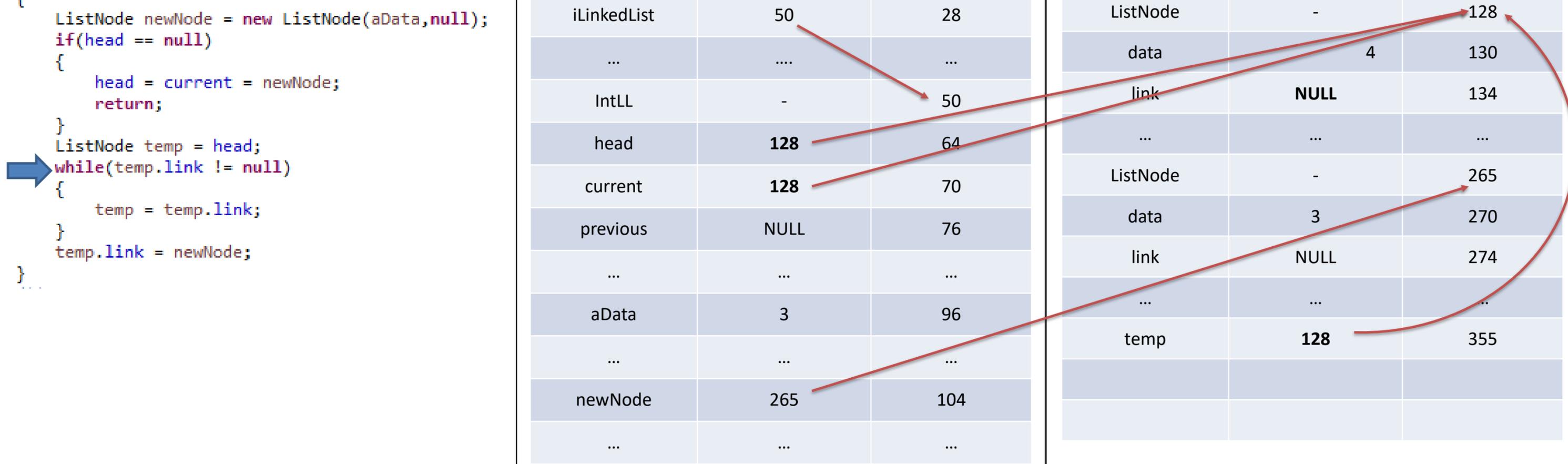
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| temp       | 128      | 355          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

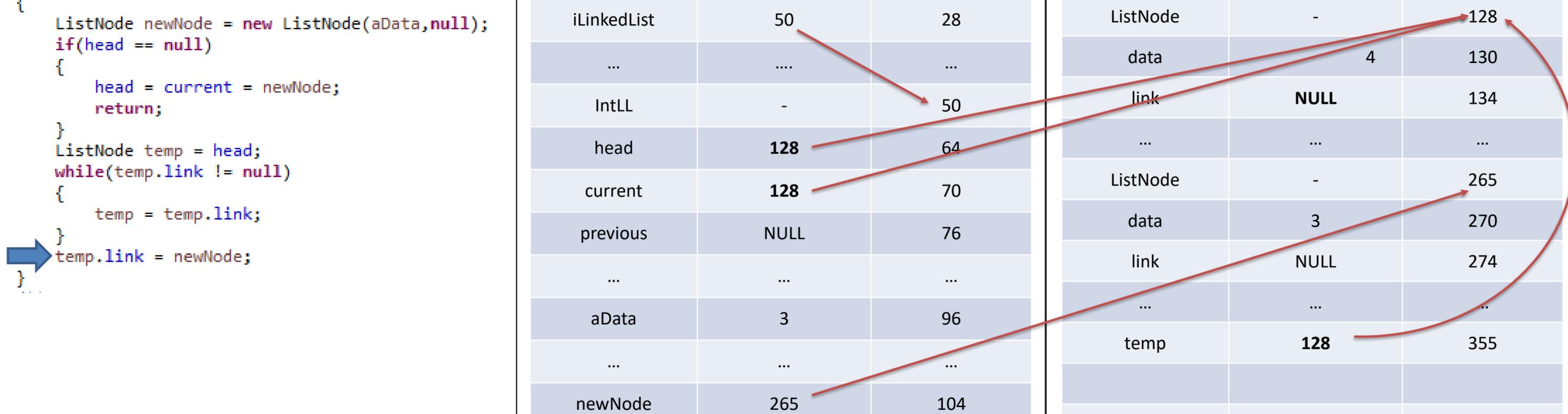
public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | NULL     | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| temp       | 128      | 355          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

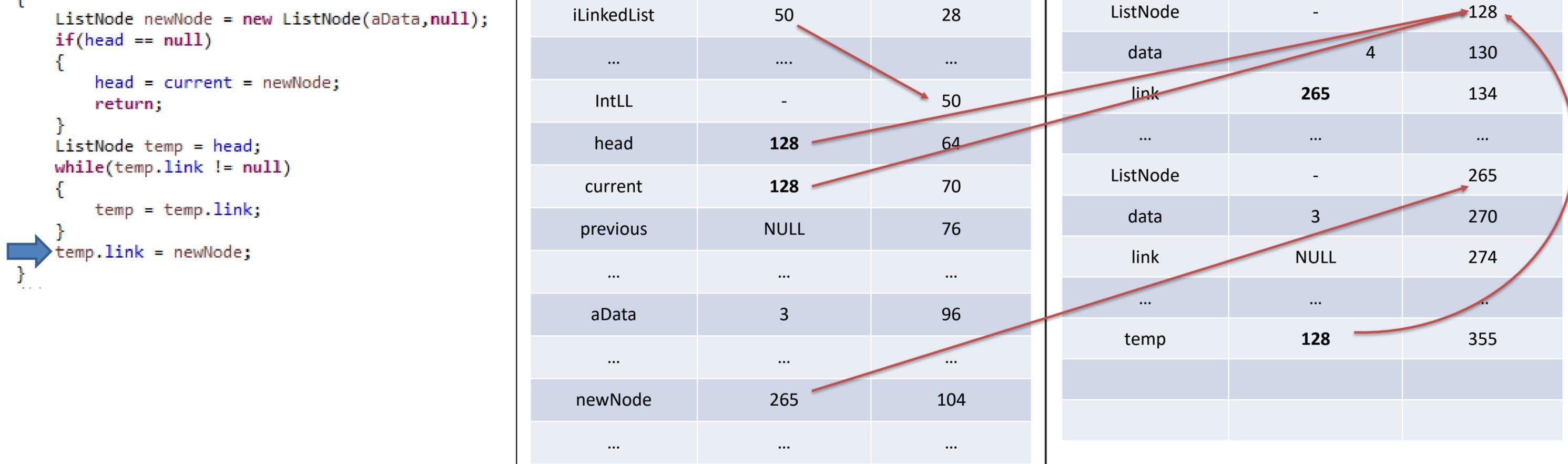
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| temp       | 128      | 355          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

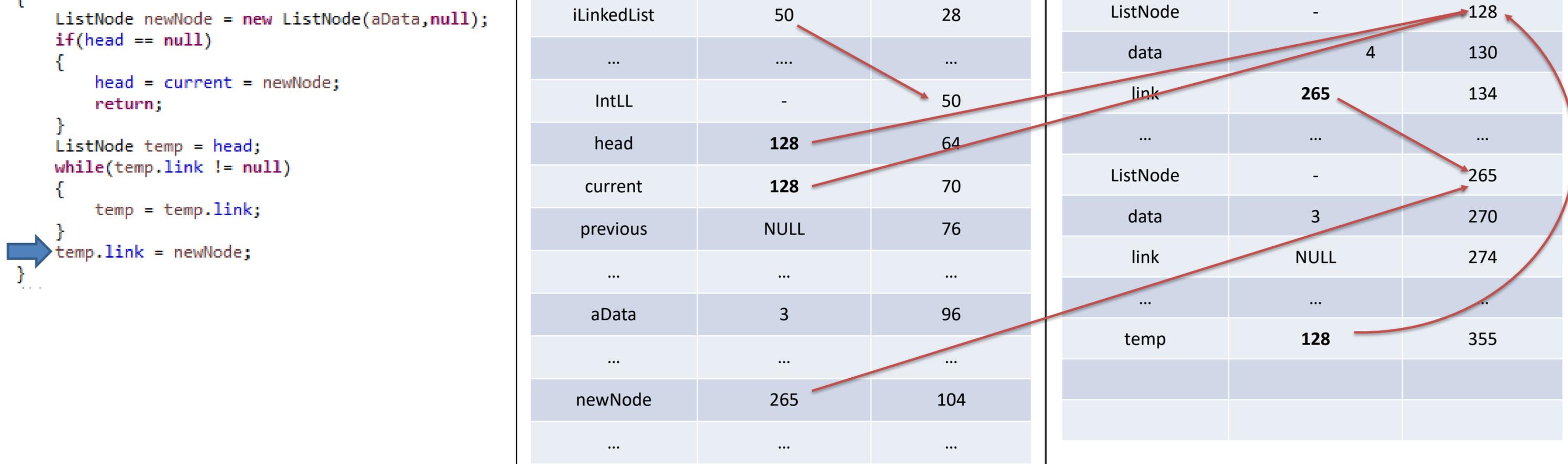
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
| aData       | 3        | 96           |
| ...         | ...      | ...          |
| newNode     | 265      | 104          |
| ...         | ...      | ...          |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
| temp       | 128      | 355          |
| ...        | ...      | ...          |



# Adding in Memory

3

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

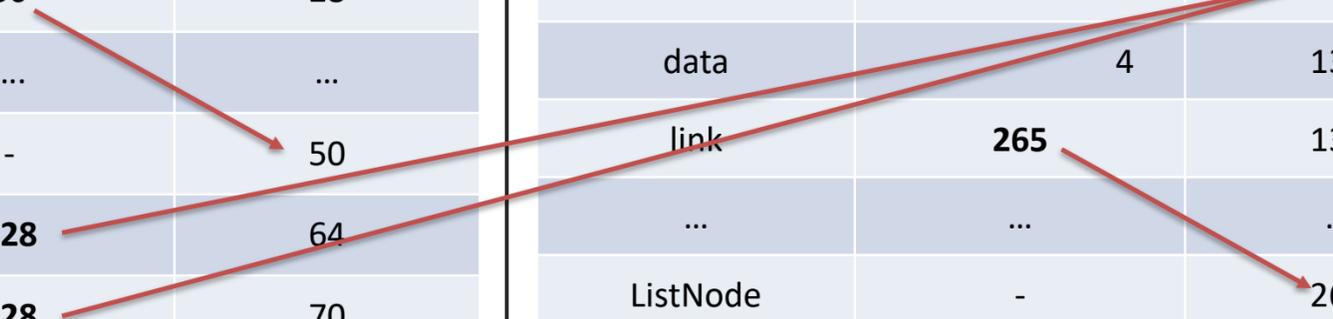
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |



# Adding in Memory

2

```

public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}

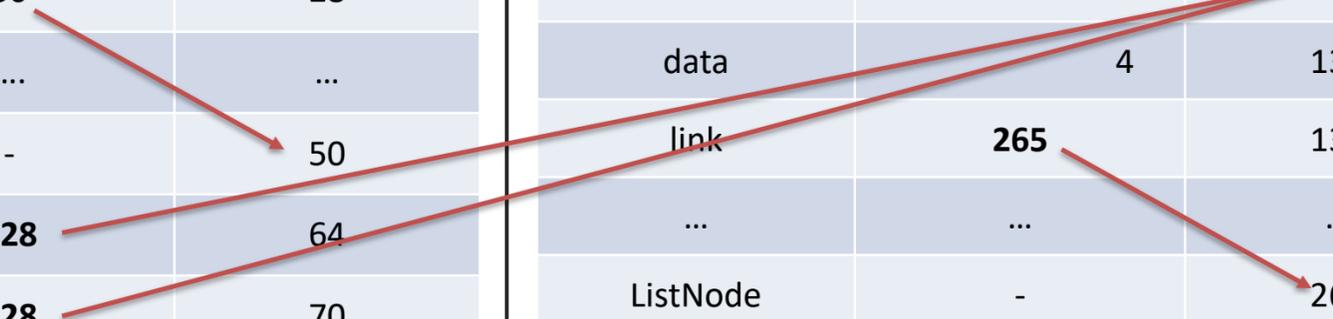
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

## More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | NULL     | 274          |
| ...        | ...      | ...          |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |



# Adding in Memory

3

```

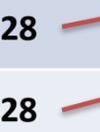
public void add(int aData)
{
    ListNode newNode = new ListNode(aData,null);
    if(head == null)
    {
        head = current = newNode;
        return;
    }
    ListNode temp = head;
    while(temp.link != null)
    {
        temp = temp.link;
    }
    temp.link = newNode;
}
    
```

## Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 128      | 70           |
| previous    | NULL     | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

## More Memory

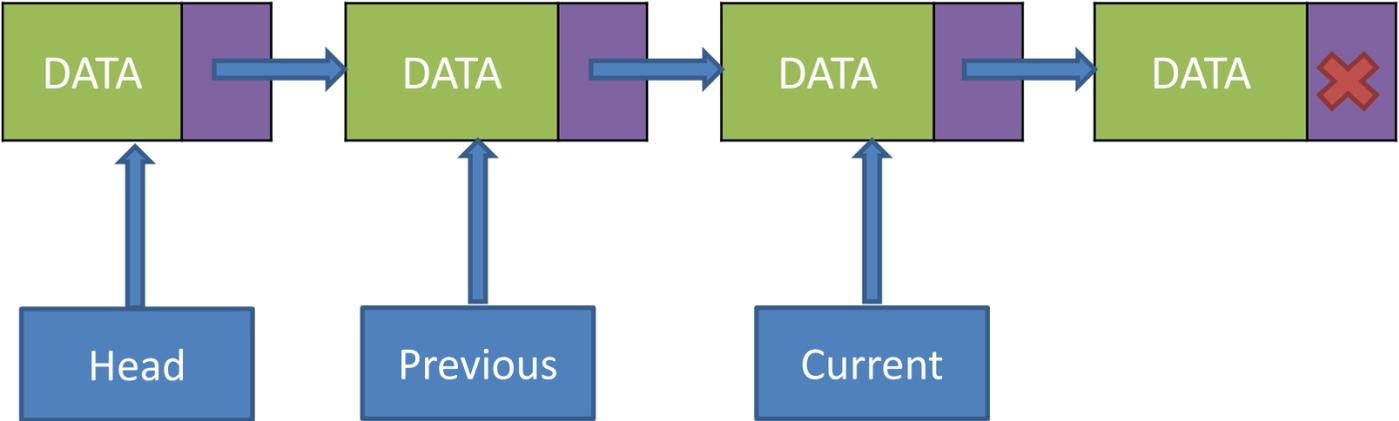
| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Adding After Current

- Create a new Node with the given Data
- Set new Node's Link to Current's Link
- Point Current's Link to the new Node

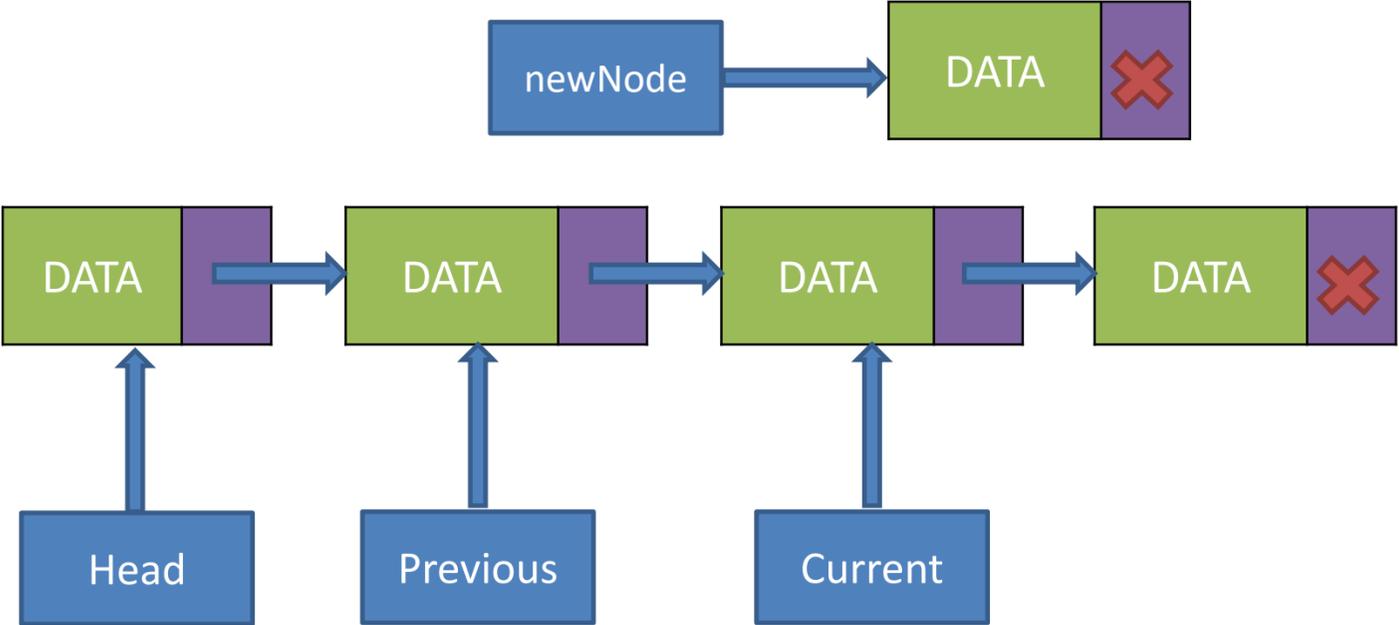
## Concept



# Adding After Current

- Create a new Node with the given Data
- Set new Node's Link to Current's Link
- Point Current's Link to the new Node

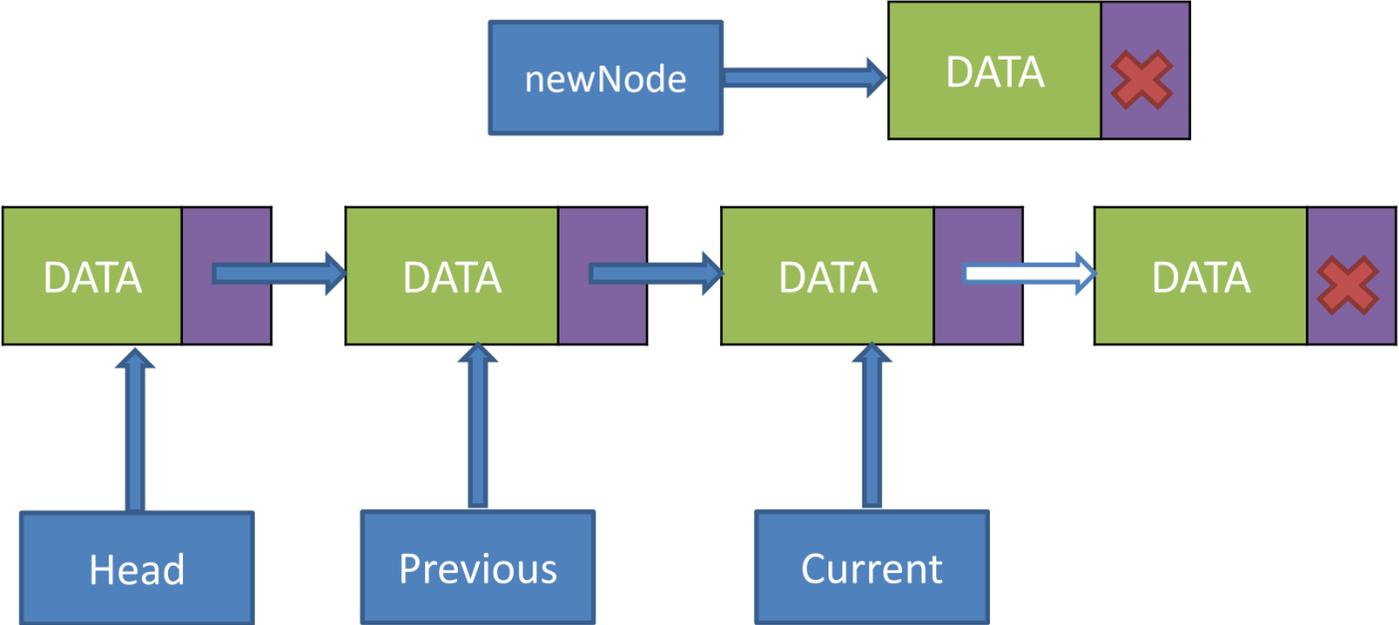
## Concept



# Adding After Current

- Create a new Node with the given Data
- Set new Node's Link to Current's Link
- Point Current's Link to the new Node

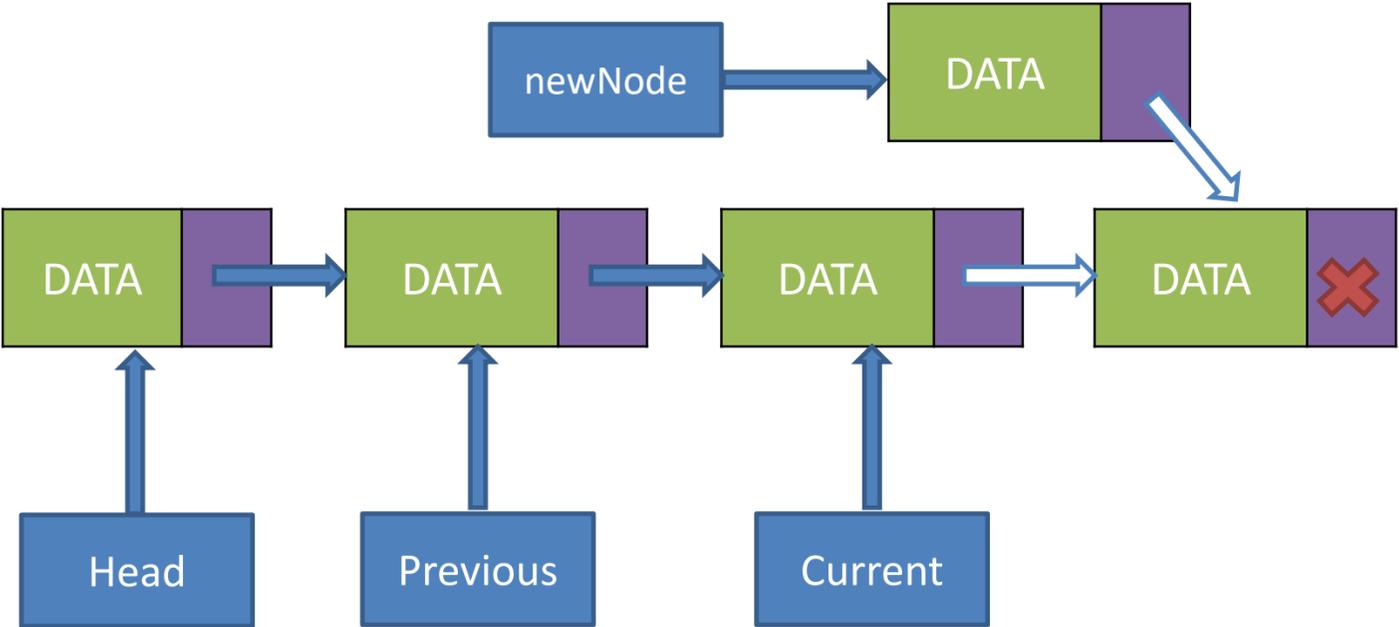
## Concept



# Adding After Current

- Create a new Node with the given Data
- Set new Node's Link to Current's Link
- Point Current's Link to the new Node

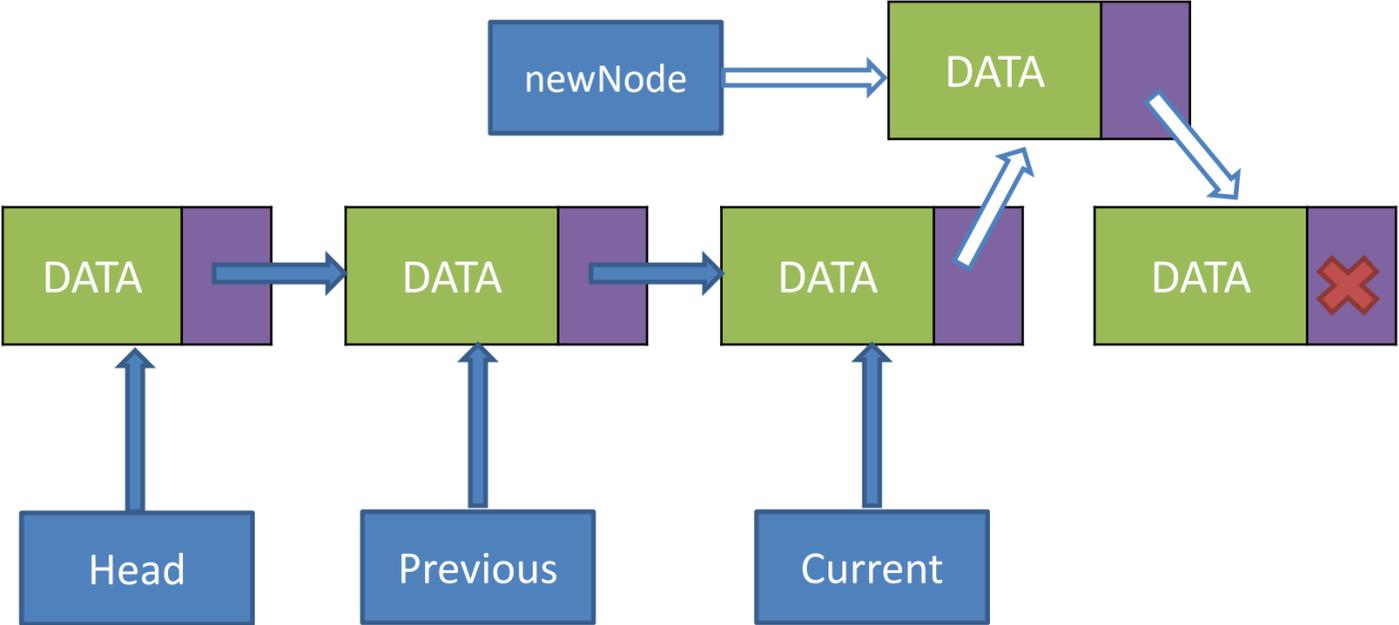
## Concept



# Adding After Current

- Create a new Node with the given Data
- Set new Node's Link to Current's Link
- Point Current's Link to the new Node

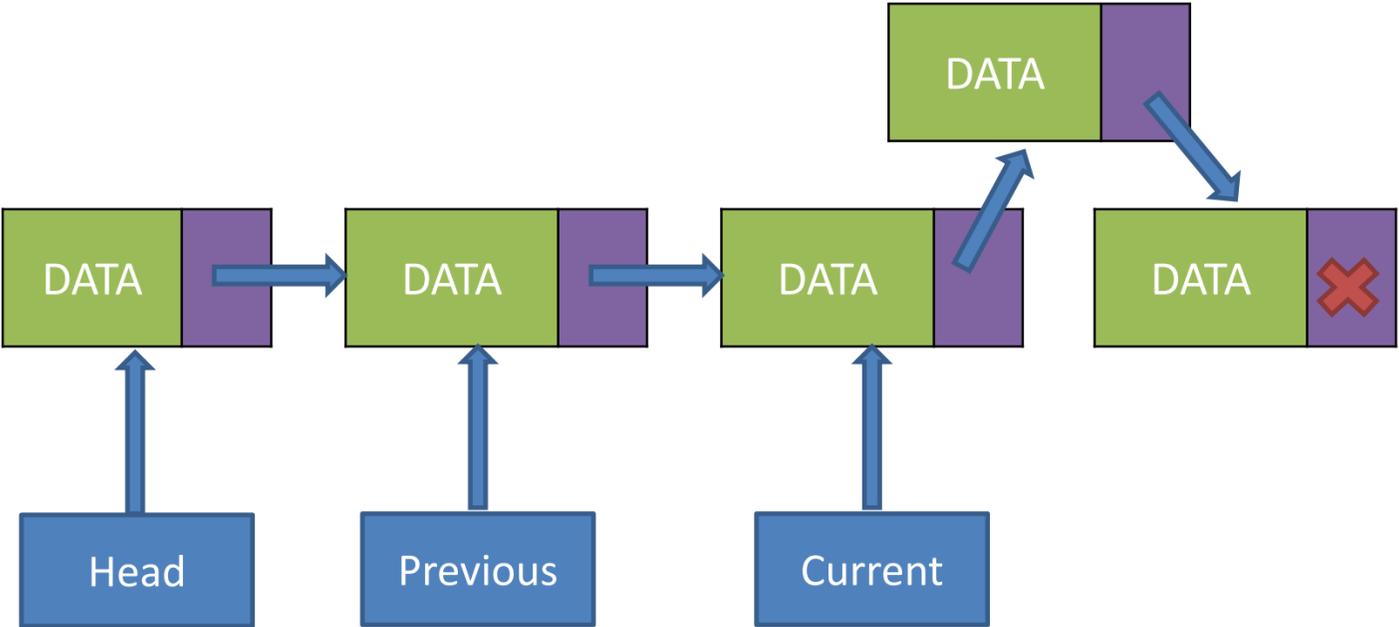
## Concept



# Adding After Current

- Create a new Node with the given Data
- Set new Node's Link to Current's Link
- Point Current's Link to the new Node

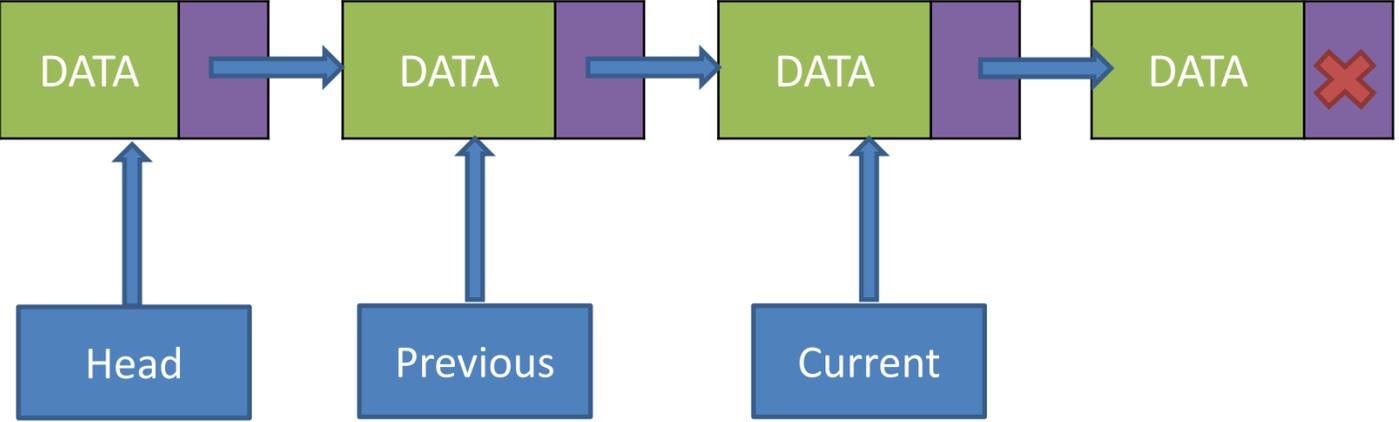
## Concept



# Removing Current

- If the Current is referencing the Head
  - Move Head and Current forward one node
- Set the Previous' Link to Current's Link
- Move Current Forward

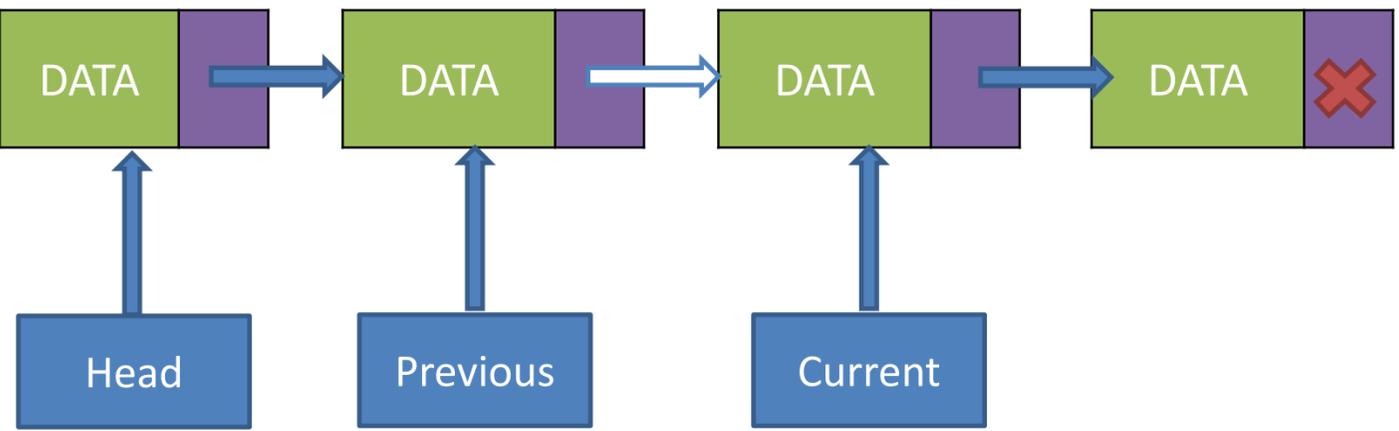
## Concept



# Removing Current

- If the Current is referencing the Head
  - Move Head and Current forward one node
- Set the Previous' Link to Current's Link
- Move Current Forward

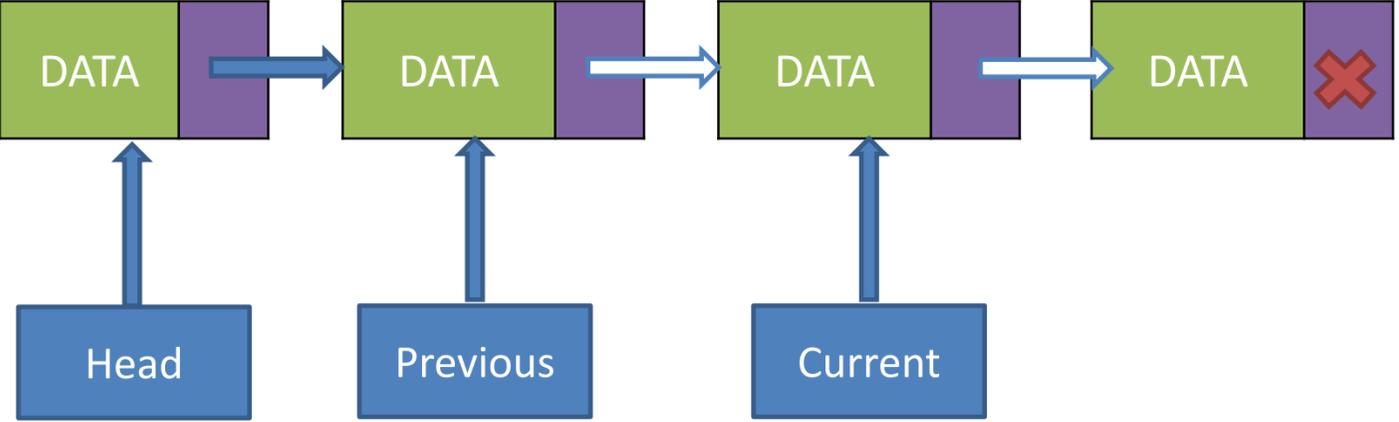
## Concept



# Removing Current

- If the Current is referencing the Head
  - Move Head and Current forward one node
- Set the Previous' Link to Current's Link
- Move Current Forward

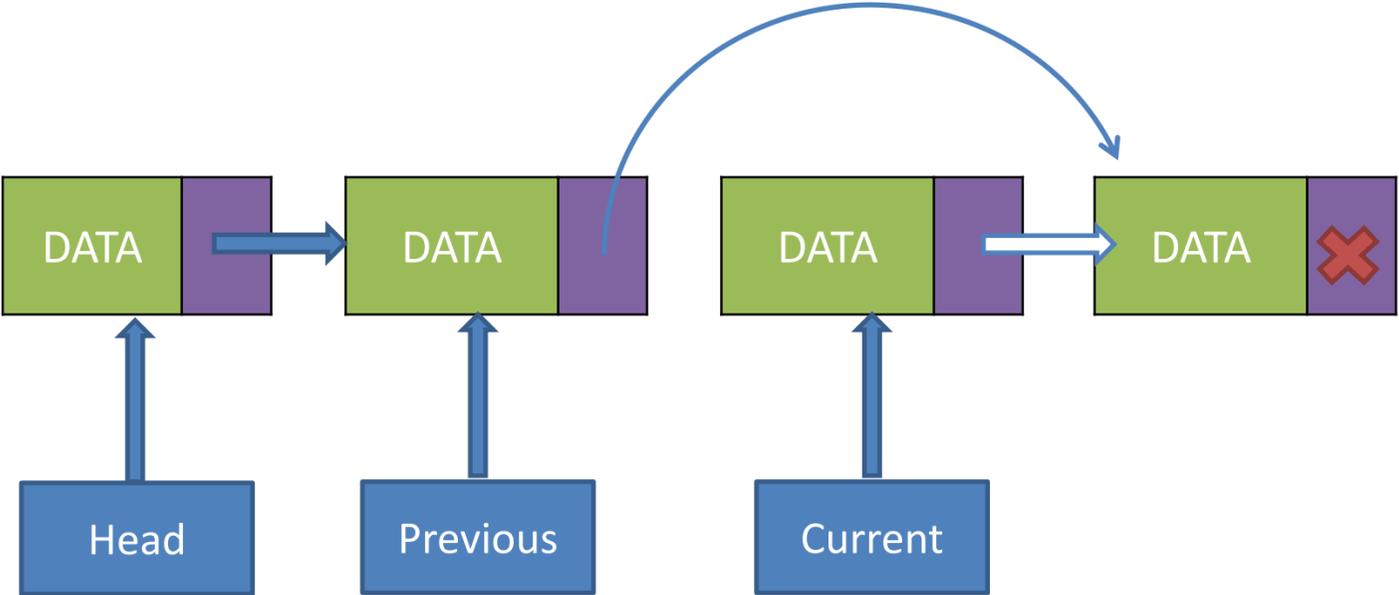
## Concept



# Removing Current

- If the Current is referencing the Head
  - Move Head and Current forward one node
- Set the Previous' Link to Current's Link
- Move Current Forward

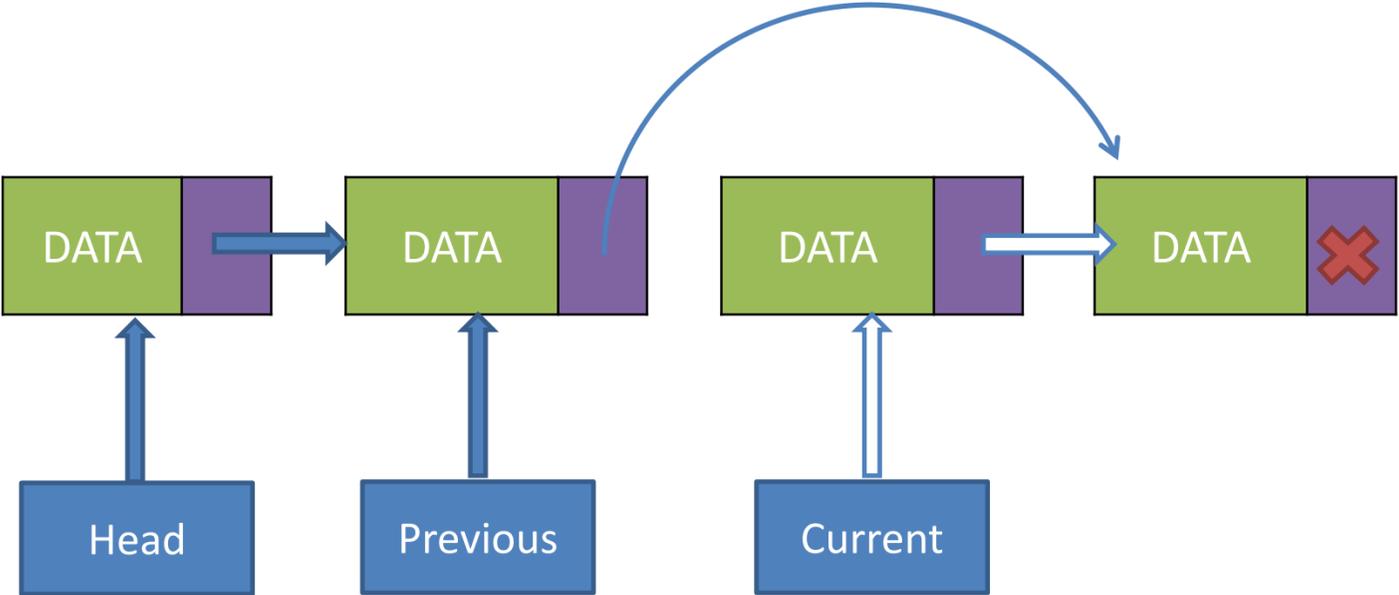
## Concept



# Removing Current

- If the Current is referencing the Head
  - Move Head and Current forward one node
- Set the Previous' Link to Current's Link
- Move Current Forward

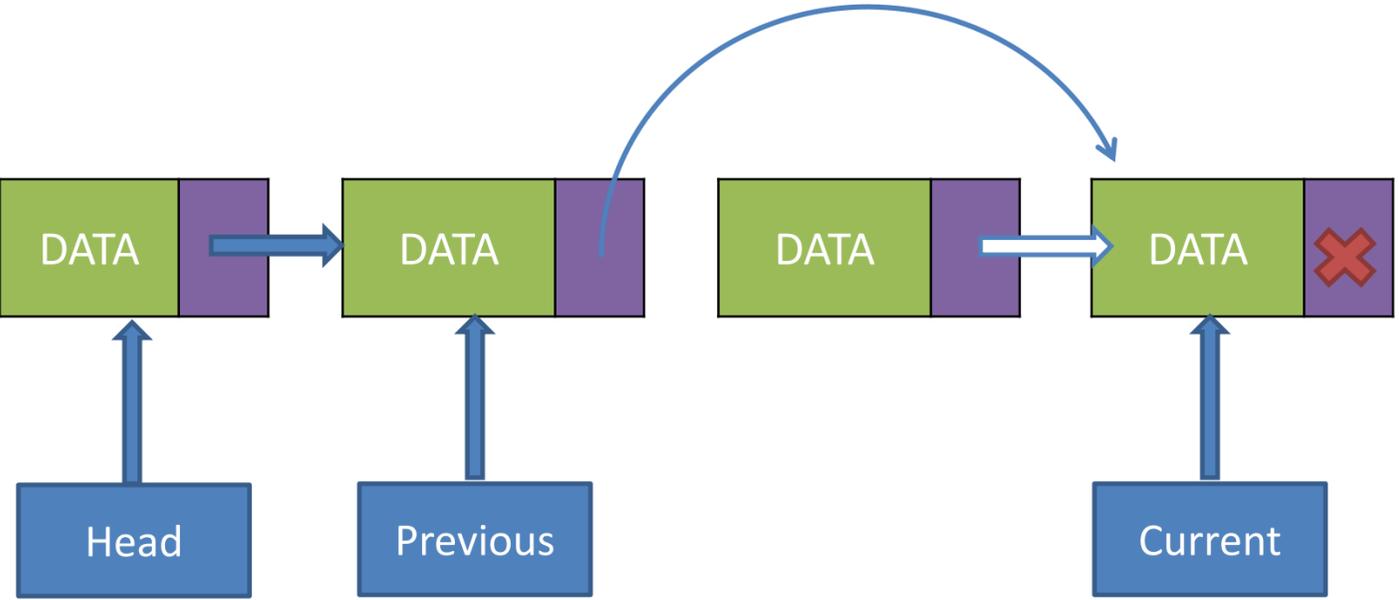
## Concept



# Removing Current

- If the Current is referencing the Head
  - Move Head and Current forward one node
- Set the Previous' Link to Current's Link
- Move Current Forward

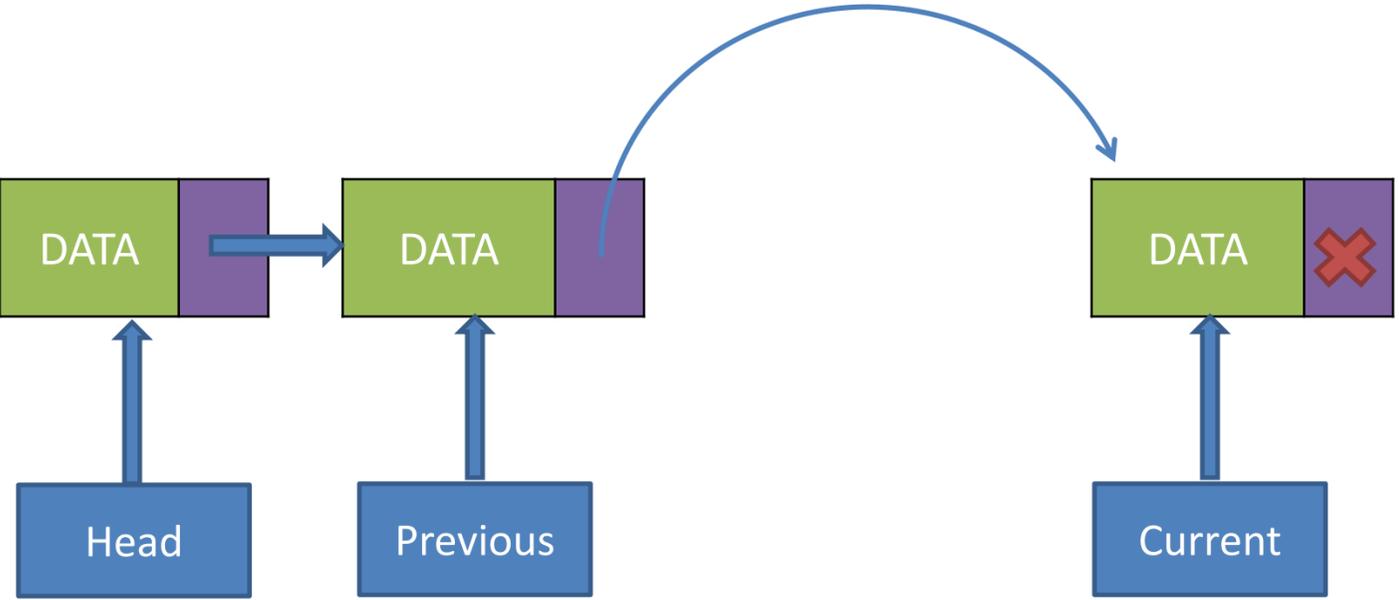
## Concept



# Removing Current

- If the Current is referencing the Head
  - Move Head and Current forward one node
- Set the Previous' Link to Current's Link
- Move Current Forward

## Concept



# Removing Current in Memory

```

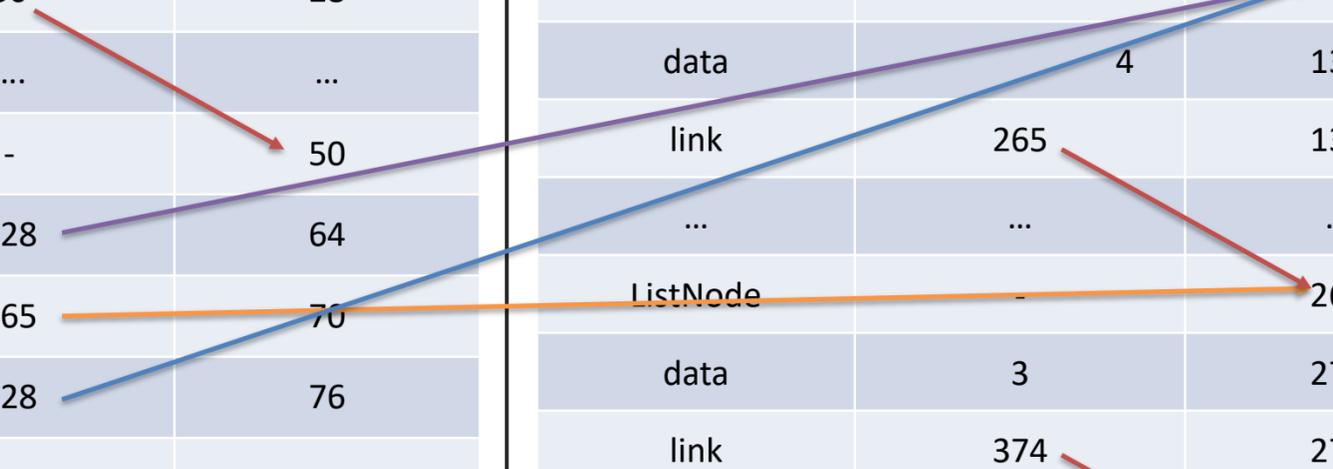
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
        previous.link = current.link;
        current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ....     | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 265      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Removing Current in Memory

```

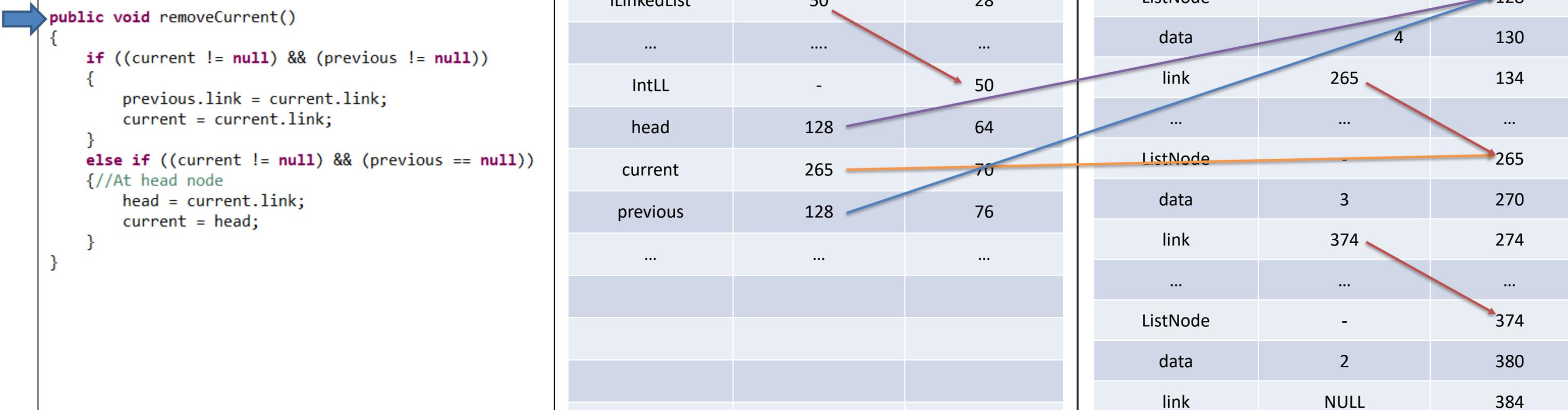
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
        previous.link = current.link;
        current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 265      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Removing Current in Memory

```

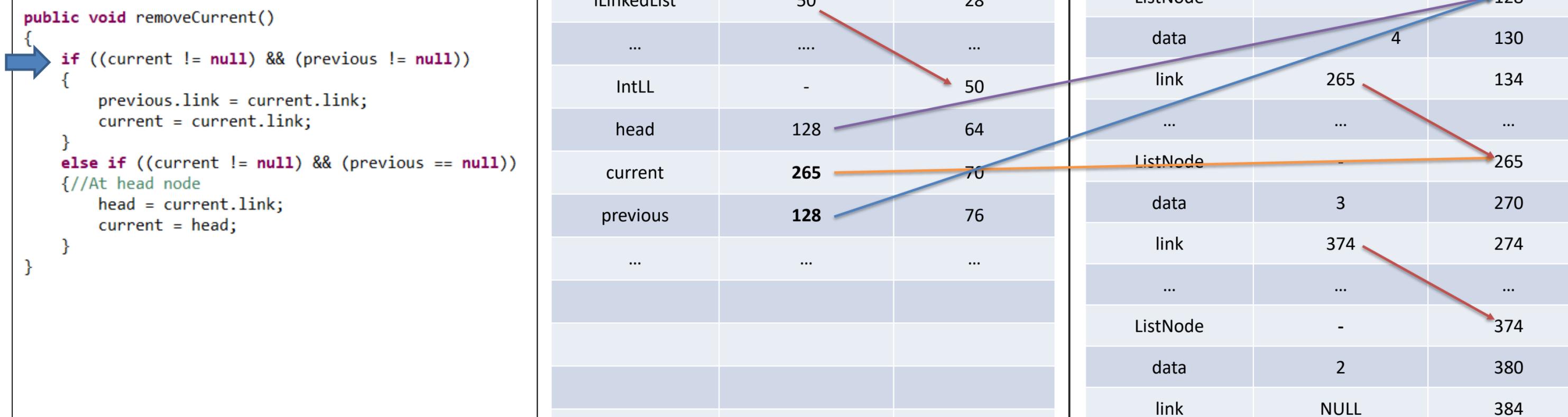
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
        previous.link = current.link;
        current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 265      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Removing Current in Memory

```

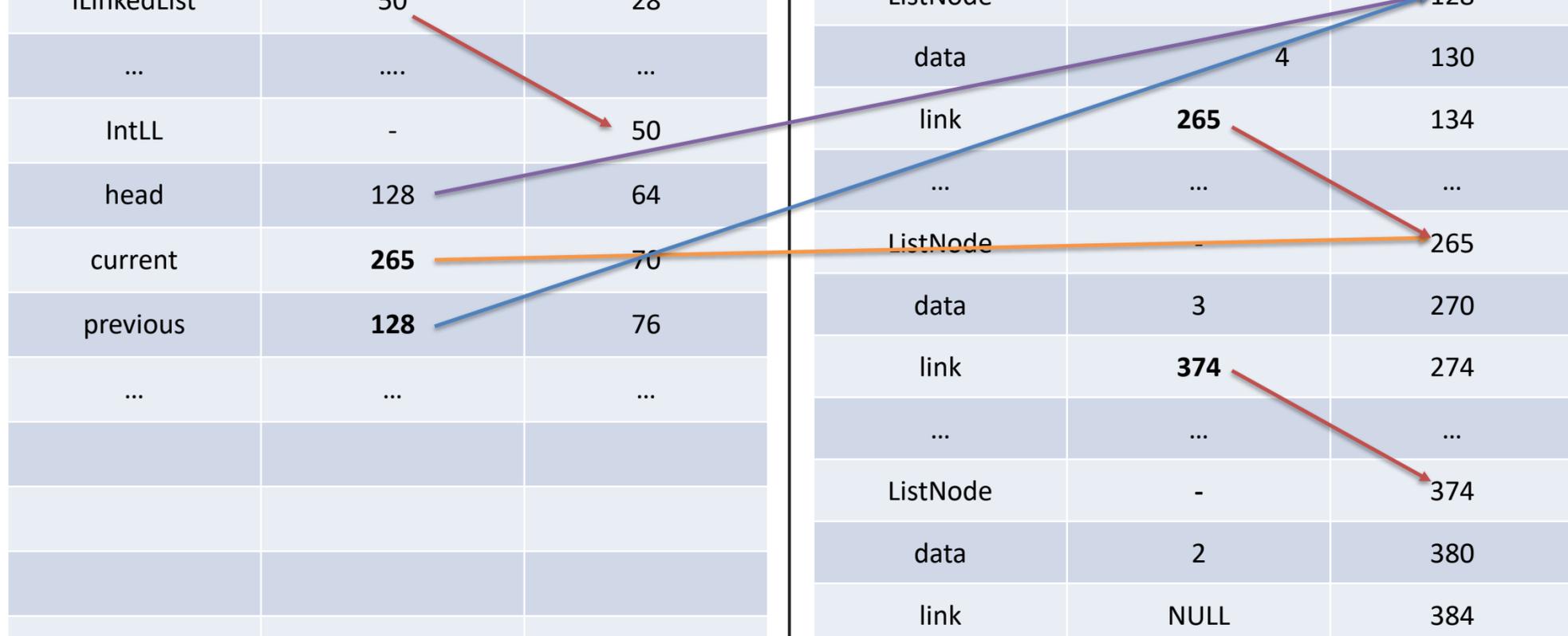
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
         previous.link = current.link;
        current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 265      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 265      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Removing Current in Memory

```

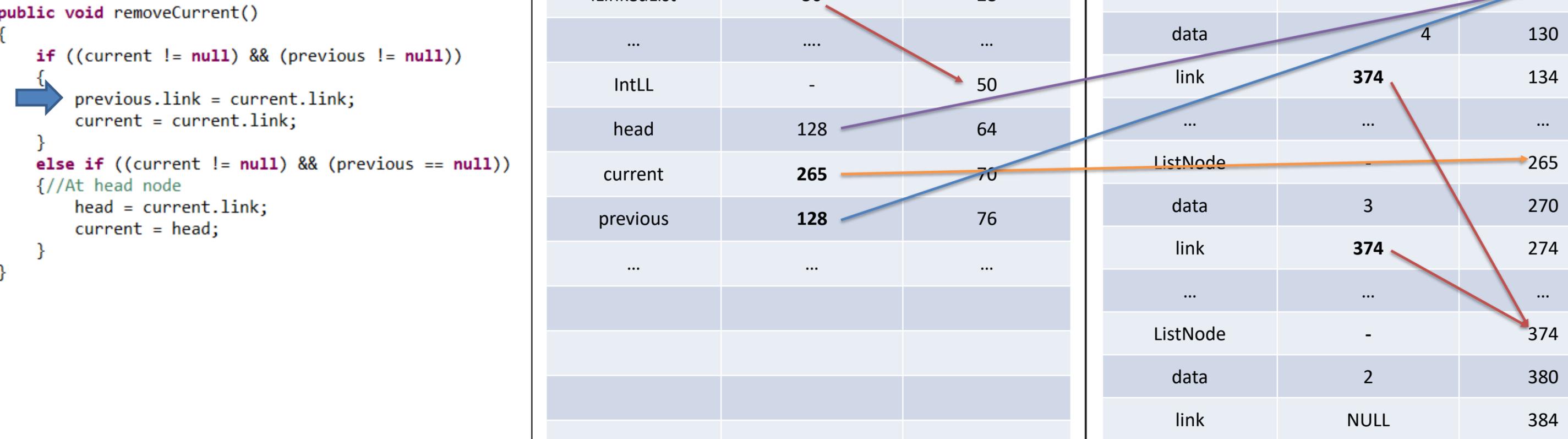
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
        previous.link = current.link;
        current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 265      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 374      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Removing Current in Memory

```

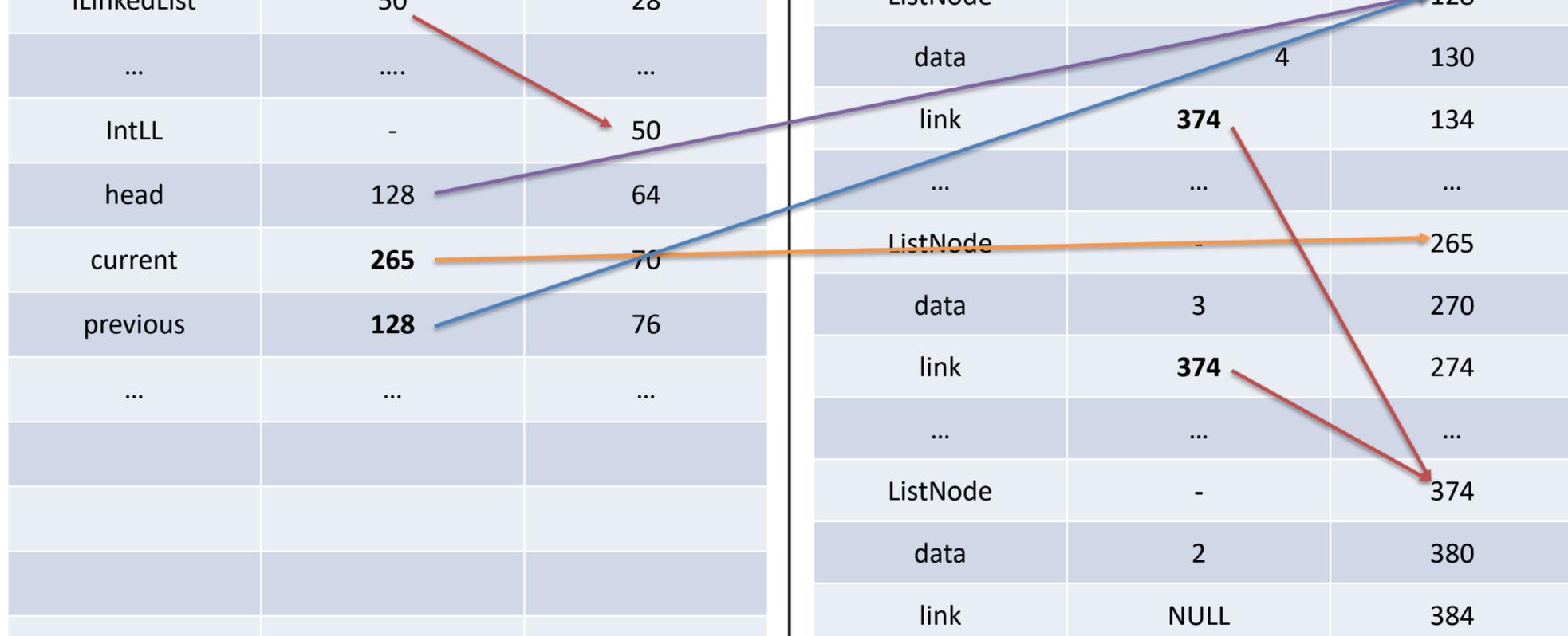
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
        previous.link = current.link;
         current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 265      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 374      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Removing Current in Memory

```

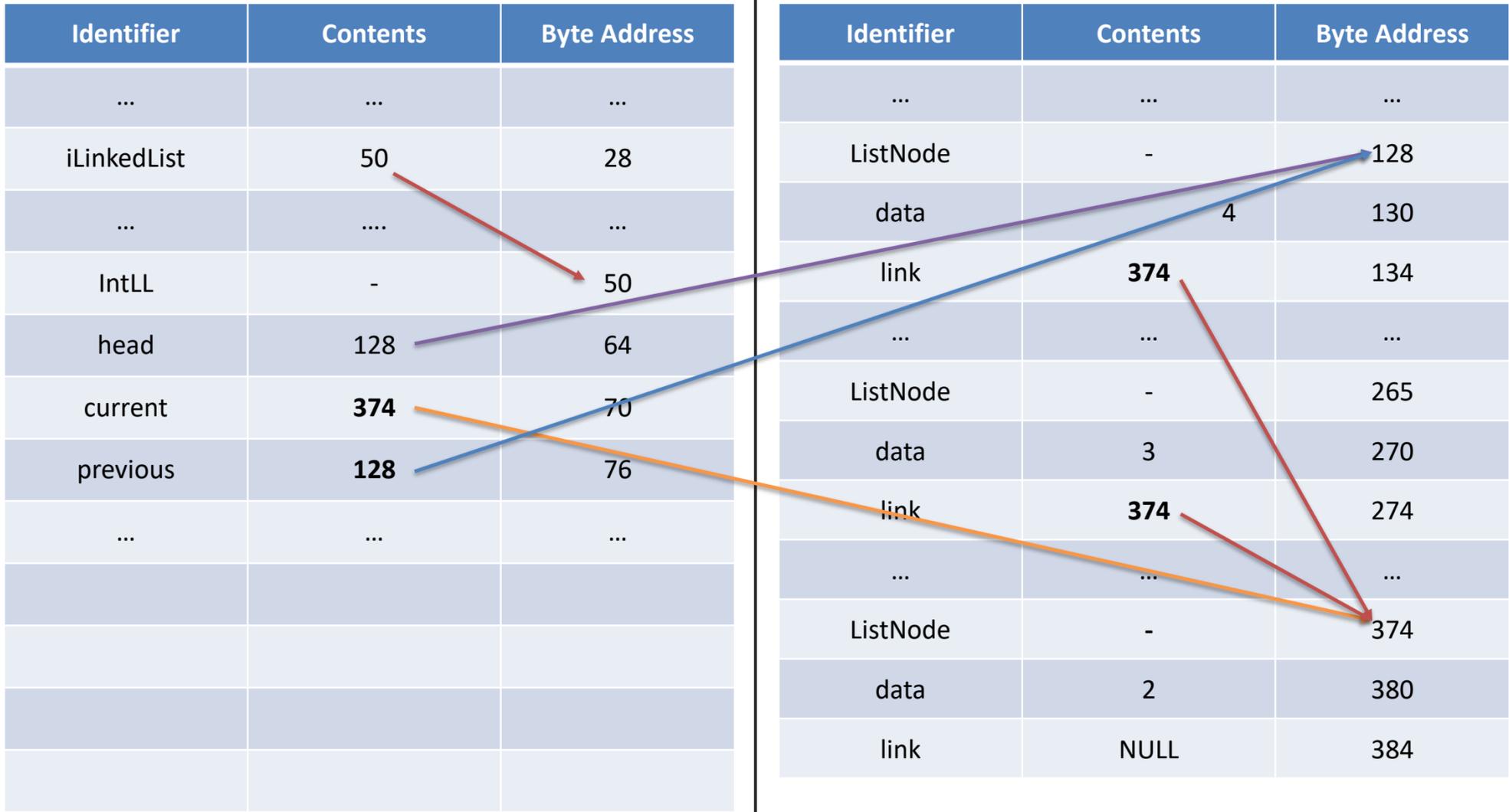
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
        previous.link = current.link;
         current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ....     | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 374      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 374      | 134          |
| ...        | ...      | ...          |
| ListNode   | -        | 265          |
| data       | 3        | 270          |
| link       | 374      | 274          |
| ...        | ...      | ...          |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Removing Current in Memory

```

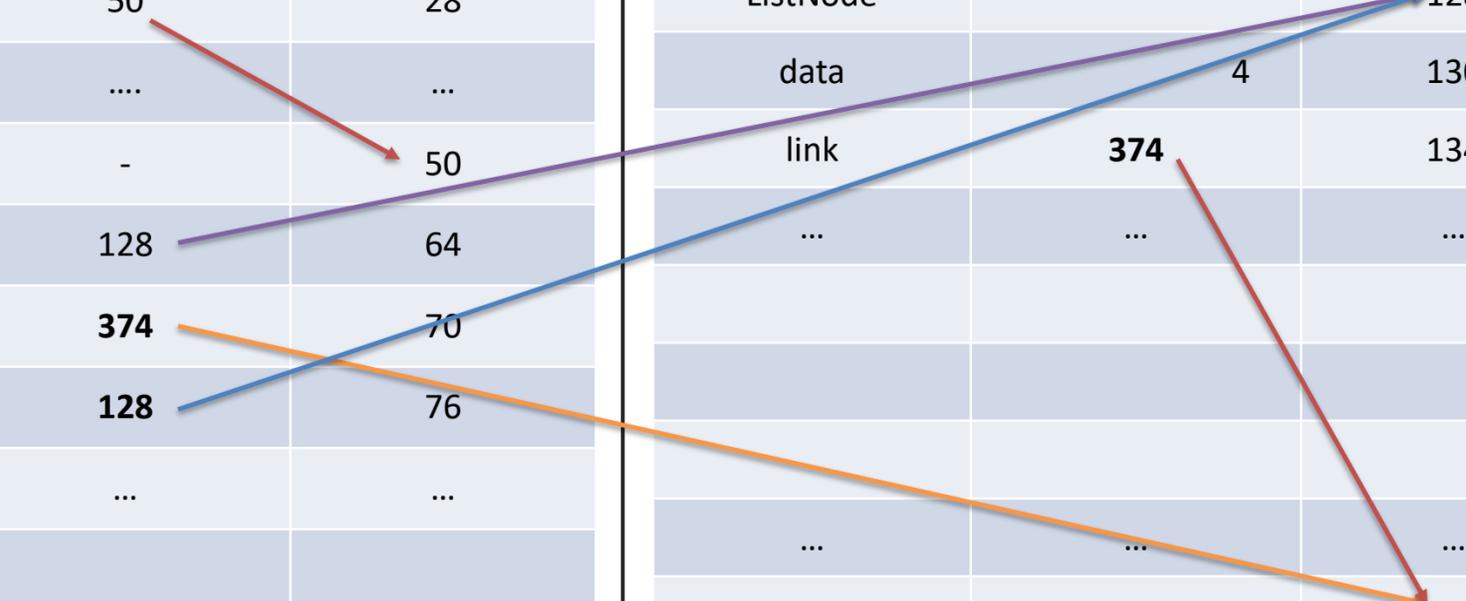
public void removeCurrent()
{
    if ((current != null) && (previous != null))
    {
        previous.link = current.link;
         current = current.link;
    }
    else if ((current != null) && (previous == null))
    { //At head node
        head = current.link;
        current = head;
    }
}
    
```

### Memory

| Identifier  | Contents | Byte Address |
|-------------|----------|--------------|
| ...         | ...      | ...          |
| iLinkedList | 50       | 28           |
| ...         | ...      | ...          |
| IntLL       | -        | 50           |
| head        | 128      | 64           |
| current     | 374      | 70           |
| previous    | 128      | 76           |
| ...         | ...      | ...          |
|             |          |              |
|             |          |              |
|             |          |              |
|             |          |              |

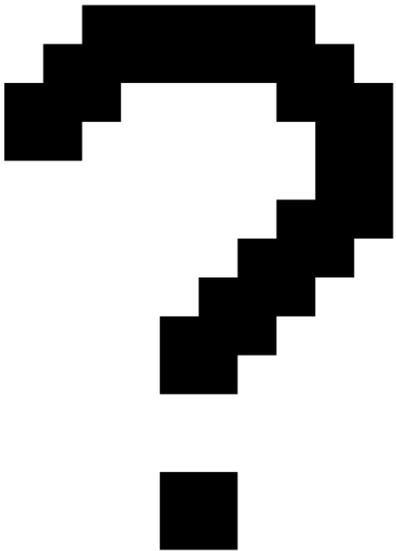
### More Memory

| Identifier | Contents | Byte Address |
|------------|----------|--------------|
| ...        | ...      | ...          |
| ListNode   | -        | 128          |
| data       | 4        | 130          |
| link       | 374      | 134          |
| ...        | ...      | ...          |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
|            |          |              |
| ListNode   | -        | 374          |
| data       | 2        | 380          |
| link       | NULL     | 384          |



# Problems

- This is only a Linked List of Integers
- How can we make this same structure without having to rewrite the code for every type?



# Linking Structures

Part 01