자료구조: 2022년 1학기 [강의]

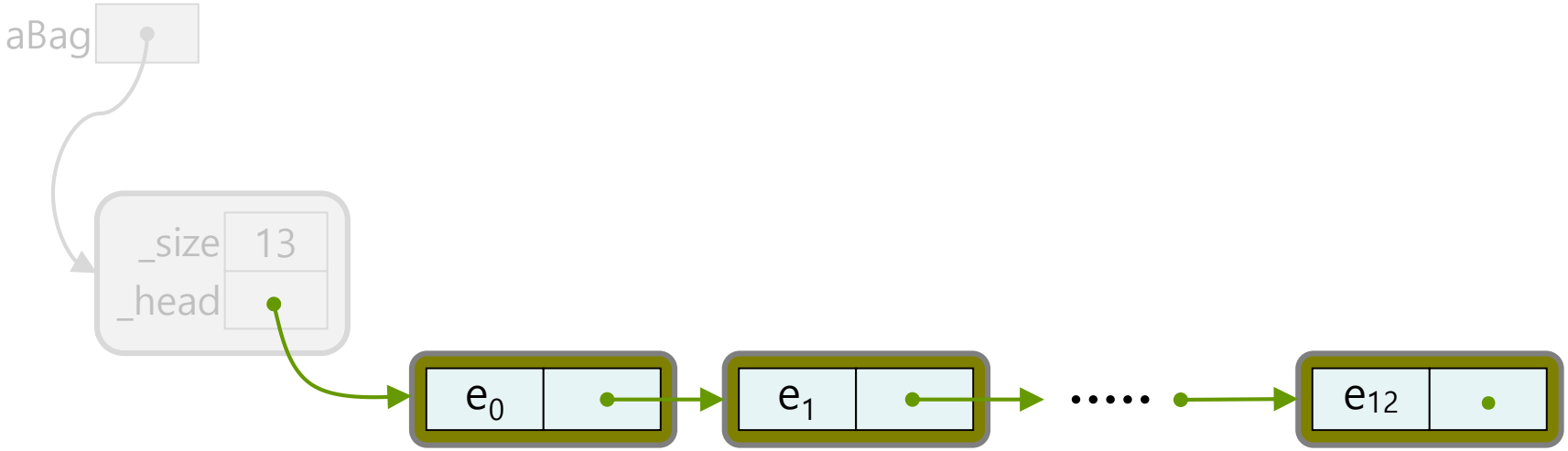
Linked Bag (1)

강지훈
jhkang@cnu.ac.kr
충남대학교 컴퓨터융합학부

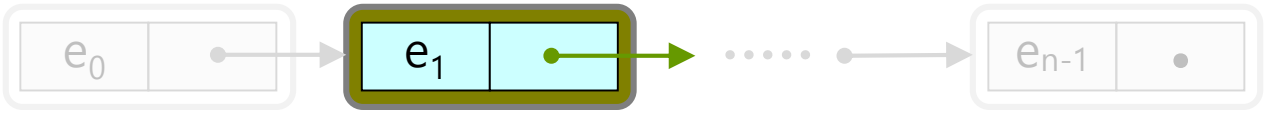
연결 체인 (Linked Chain)

- Class “Bag” 구현에 사용 -

Linked Chain

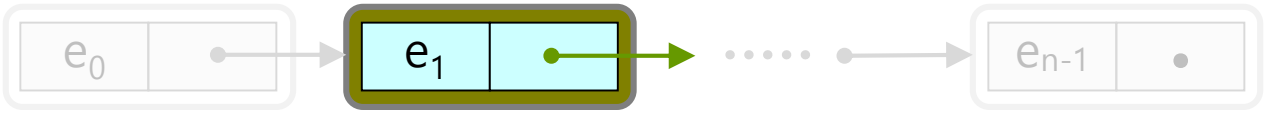


Class “LinkedNode”



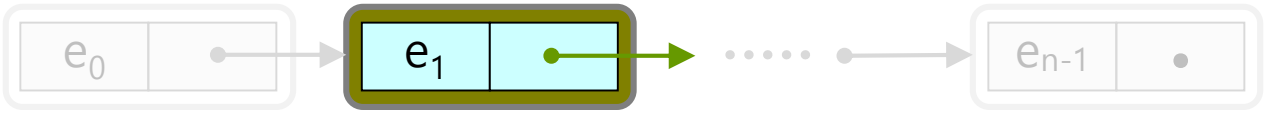
LinkedNode 의 공개함수

- LinkedNode 객체 사용법을 Java 로 구체적으로 표현
  - public LinkedNode () { }
  - public LinkedNode (E givenElement, LinkedNode<E> givenNext) { }
  - public E element () { }
  - public LinkedNode<E> next () { }
  - public void setElement (E newElement) { }
  - public void setNext (LinkedNode<E> newNext) { }



LinkedNode: 멤버 변수

```
public class  LinkedNode<E>
{
    // 비공개 인스턴스 변수
    private   E                  _element ;
    private   LinkedNode<E>      _next ;
```



LinkedNode: Getters

```
public class  LinkedNode<E>
{
    // 비공개 멤버 변수
    ……

    // Getters
    public E  element ( )
    {
        return  this._element ;
    }

    public LinkedNode<E>  next ( )
    {
        return  this._next ;
    }
```

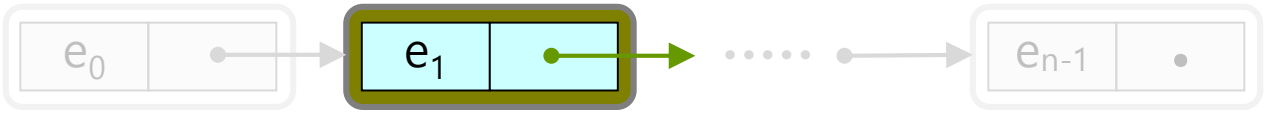
LinkedNode: Setters

```
public class  LinkedNode<E>
{
    // 비공개 멤버 변수
    ……

    // Getters
    ……

    // Setters
    public void  setElement (E newElement)
    {
        this._element = newElement ;
    }

    public void  setNext (LinkedNode<E> newNext)
    {
        this._next =  newNext ;
    }
```

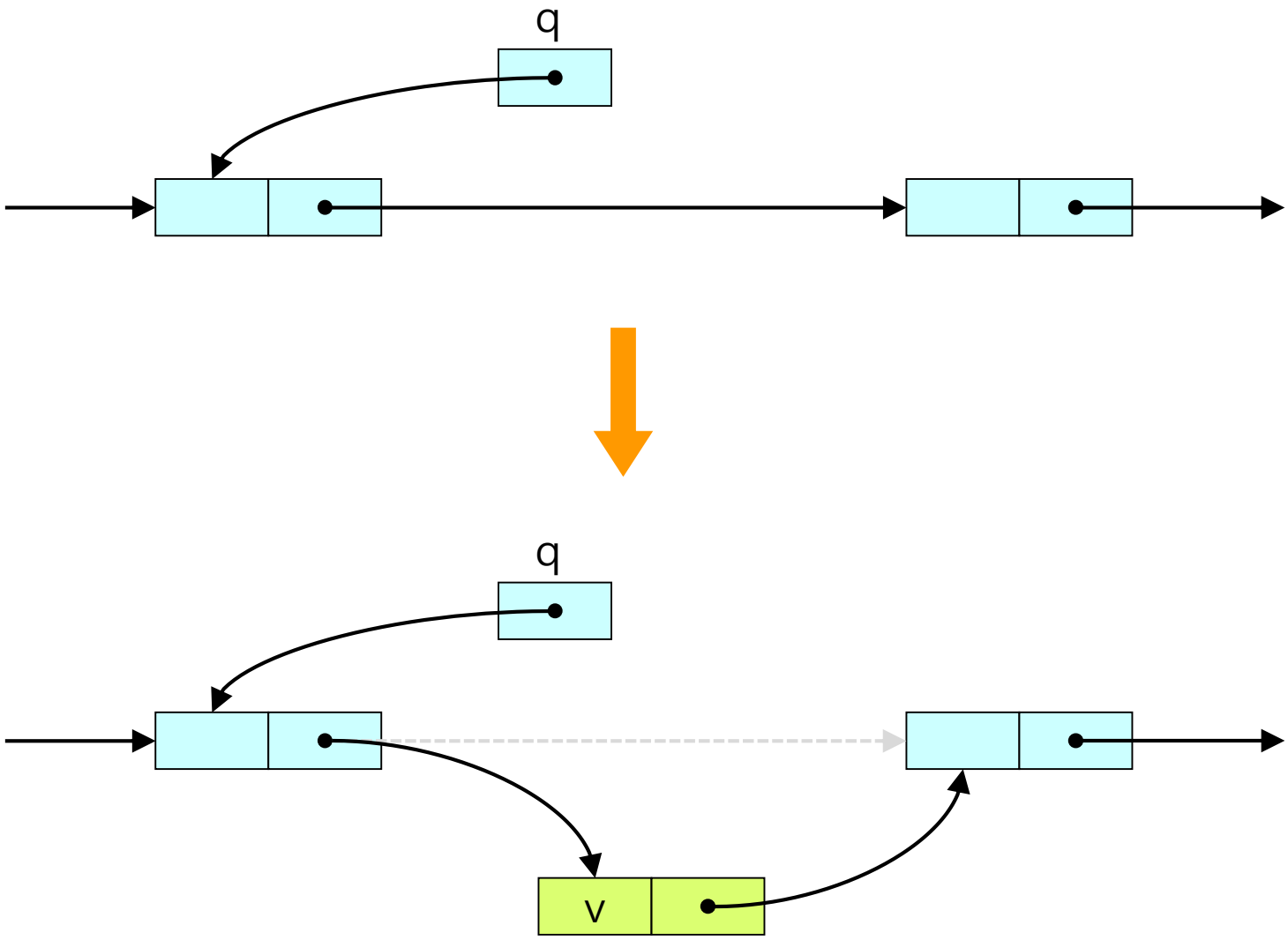
LinkedNode: 생성자

```
public class  LinkedNode<E>
{
    // 비공개 멤버 변수
    ……

    // 생성자
    public LinkedNode ( )
    {
        this.setElement (null) ;
        this.setNext (null) ;
    }

    public LinkedNode (E givenElement, LinkedNode<E> givenNext)
    {
        this.setElement (givenElement) ;
        this.setNext (givenNext) ;
    }
```

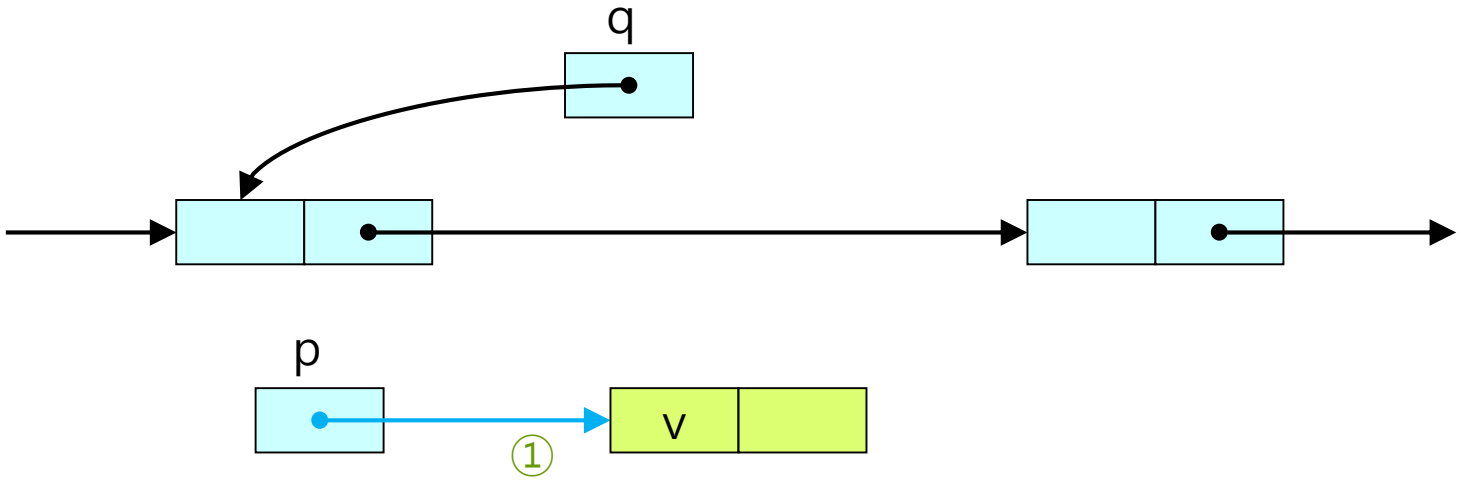
연결 체인에서의 노드의 삽입과 삭제



노드의 삽입 [1]



노드의 삽입 [2]



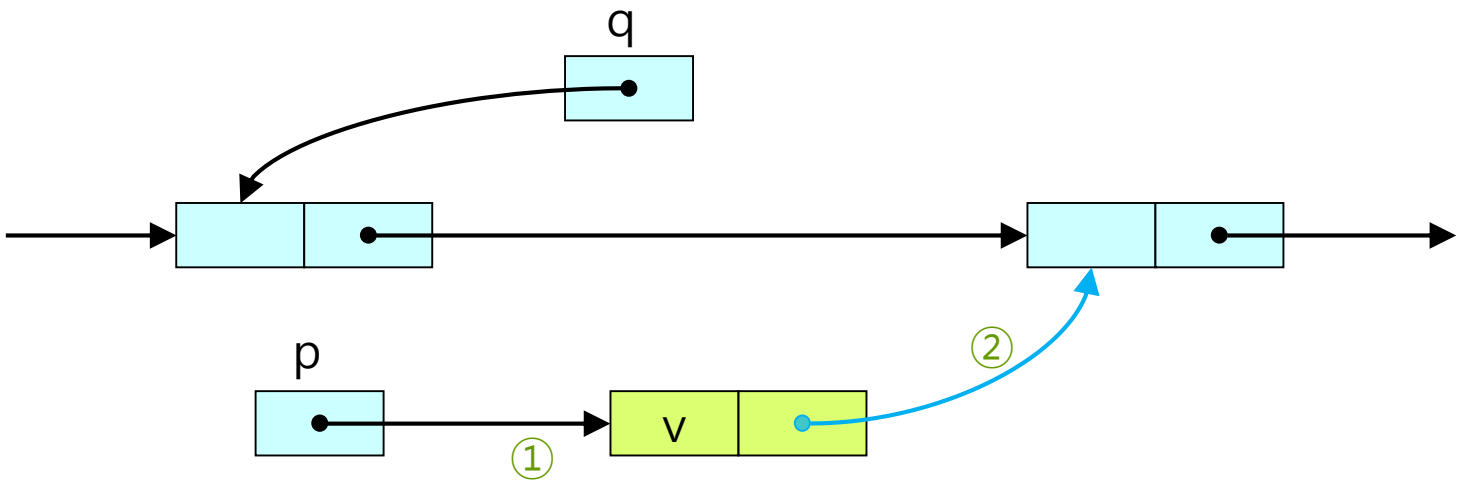
① p = new LinkedNode<E>() ;
p.setElement(v) ;

② p.setNext(q.next()) ;

③ q.setNext(p) ;

```
public class  LinkedNode<E>
{
    // 비공개 인스턴스 변수
    private  E  _element ;
    private  LinkedNode<E>  _next ;
```

노드의 삽입 [3]



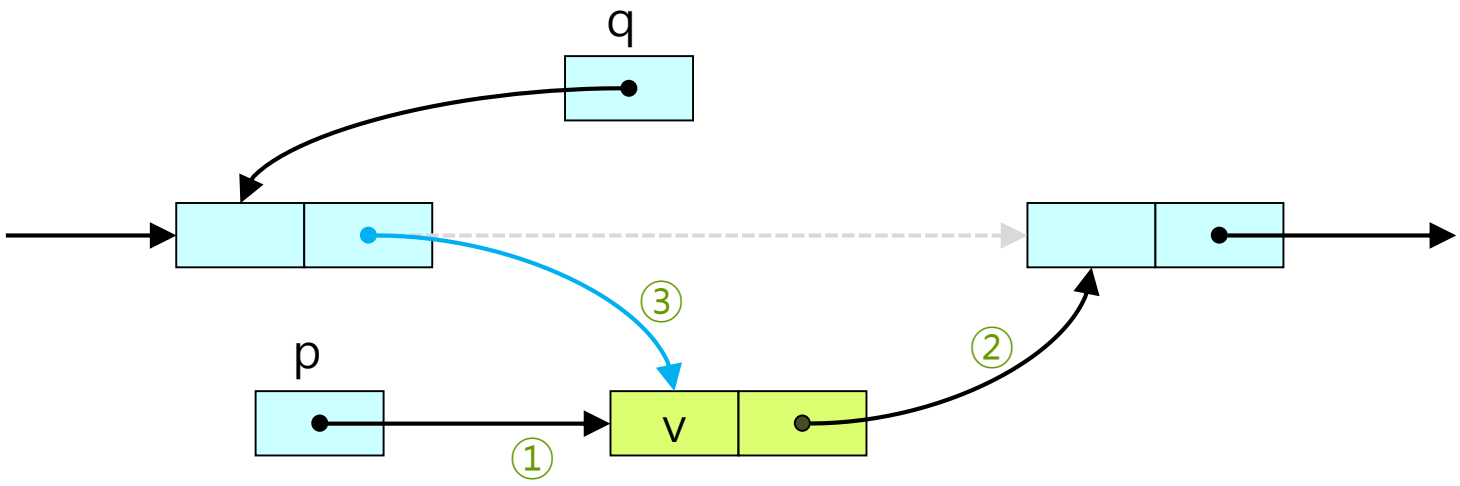
① p = new LinkedNode<E>() ;
p.setElement(v) ;

② p.setNext(q.next()) ;

③ q.setNext(p) ;

```
public class  LinkedNode<E>
{
    // 비공개 인스턴스 변수
    private  E  _element ;
    private  LinkedNode<E>  _next ;
```

노드의 삽입 [4]



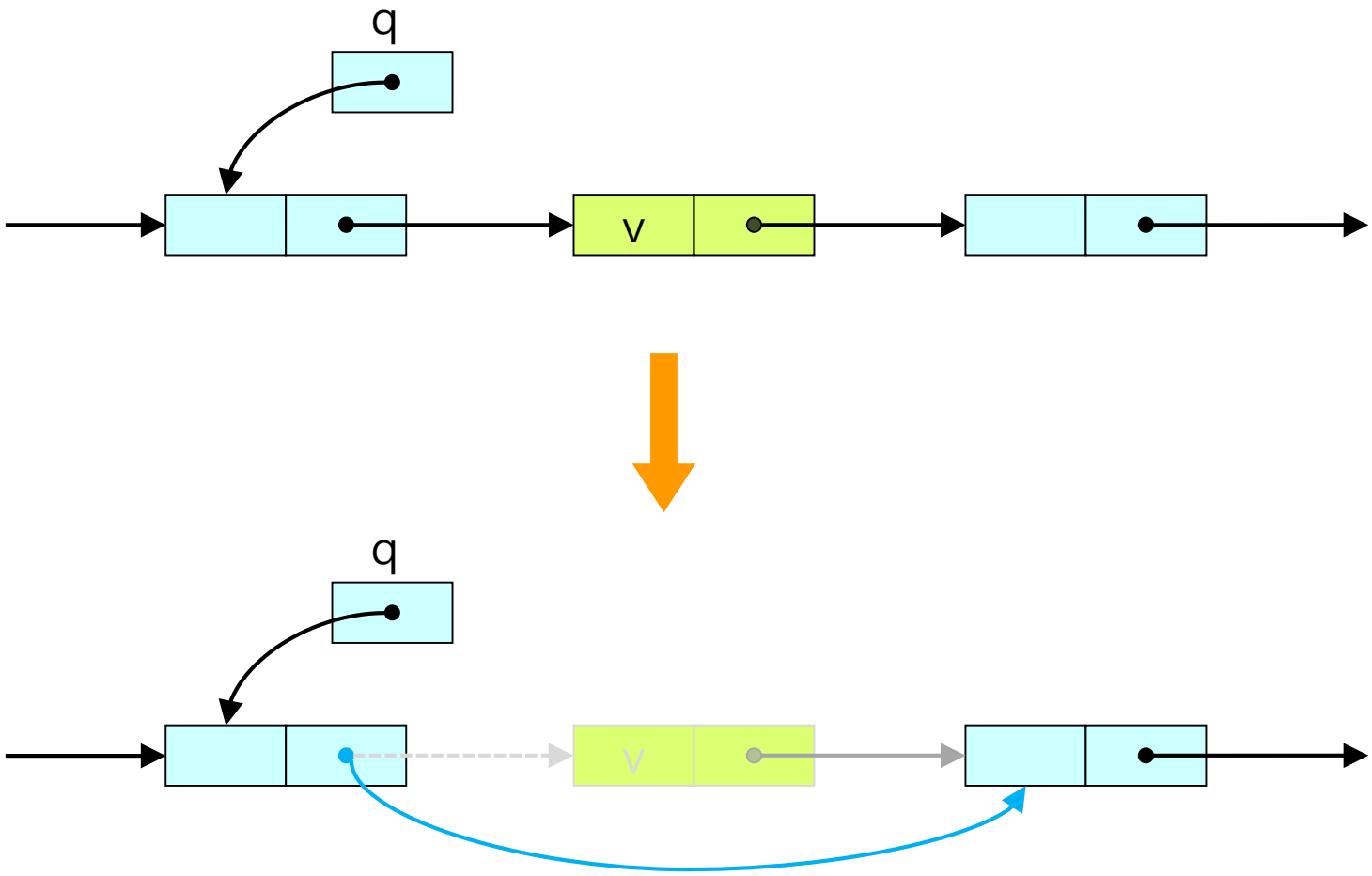
① p = new LinkedNode<E>() ;
p.setElement(v) ;

② p.setNext(q.next()) ;

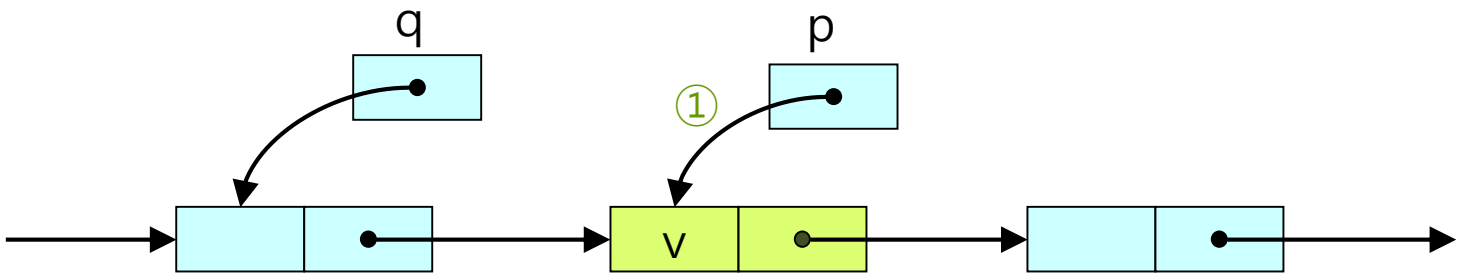
③ q.setNext(p) ;

```
public class  LinkedNode<E>
{
    // 비공개 인스턴스 변수
    private  E  _element ;
    private  LinkedNode<E>  _next ;
```

노드의 삭제 [1]



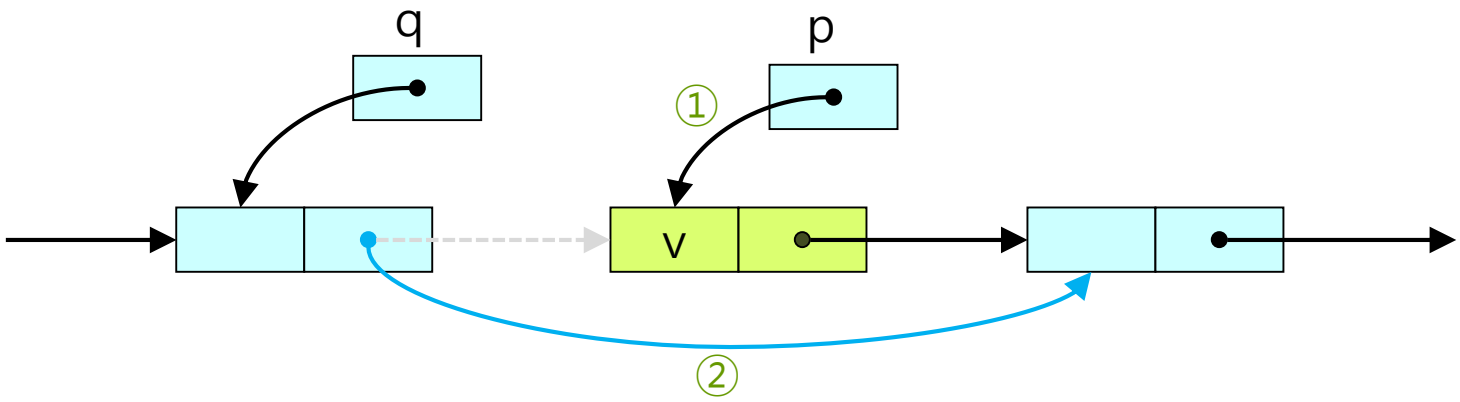
노드의 삭제 [2]



① p = q.next() ;

② q.setNext(p.next()) ;
// (또는) q.setNext((q.next()).next()) ;

노드의 삭제 [3]



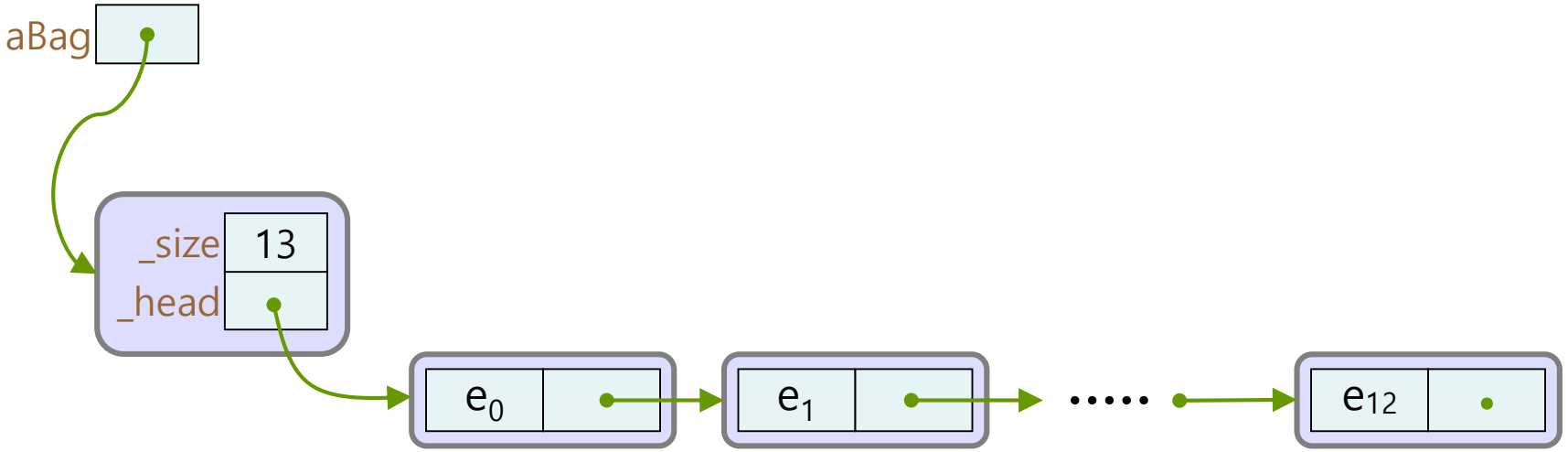
① p = q.next() ;

② q.setNext(p.next()) ;
// (또는) q.setNext((q.next()).next()) ;

Class “LinkedBag”

"LinkedBag" as a Bag

- 추상적인 Bag 을 Linked Chain 를 이용하여 구현
  - LinkedBag<Coin> aBag = new LinkedBag<Coin>() ;
    …… // doing some operations for aBag



LinkedBag<E>의 공개함수

- LinkedBag 객체 사용법을 Java 로 구체적으로 표현
  - public LinkedBag() { }
  - public int size () { }
  - public boolean isEmpty () { }
  - public boolean isFull () { }
  - public boolean doesContain (E anElement) { }
  - public int frequencyOf (E anElement) { }
  - public Element any () { }
  - public boolean add (E anElement) { }
  - public E removeAny () { }
  - public boolean remove (E anElement) { }
  - public void clear () { }

LinkedBag의 구현: 인스턴스 변수, Getters/Setters

```
public class  LinkedBag<E>
{
    // 비공개 인스턴스 변수
    private int                  _size ;
    private LinkedNode<E> _head ; // LinkedBag 의 원소들을 담을 Linked Chain

    // Getters / Setters
    public int size() { // 공개함수
        return this._size ;
     }
    private void setSize (int newSize) {
        this._size = newSize ;
    }

    private LinkedNode<E> head() {
        return  this._head ;
    }
    private void setHead (LinkedNode<E> newHead) {
        return  this._head = newHead ;
    }
```

LinkedBag: 생성자

```
public class  LinkedBag<E>
{
    // 비공개 인스턴스 변수
    ……

    // 생성자
    public LinkedBag ( )
    {
        this.setSize (0) ;
        this.setHead (null) ;
    }
```

LinkedBag: 상태 알아보기

```
public class  LinkedBag<E>
{
    // 비공개 인스턴스 변수
    ……

    // 생성자
    ……

    // 상태 알아보기
    // public int  size () // 앞에서 이미 구현
    // {
    //     return  this._size ;
    // }

    public boolean  isEmpty ()
    {
        return  (this.size() == 0) ; // 또는 return (this.head() == null) ;
    }
```

LinkedBag: 상태 알아보기

```
public class  LinkedBag<E>
{
    // 비공개 인스턴스 변수
    ……

    // 생성자
    ……

    // 상태 알아보기
    ……

    public boolean  isFull ()
    {
        return  false ; // 원소 저장 개수에 영향을 받지 않으므로.
        // 시스템 메모리 부족 오류는 없다고 가정한다.
    }
```

LinkedBag: doesContain()

```
// 상태 알아보기
.....
public boolean  doesContain (E anElement)
{

    LinkedNode<E> currentNode = this.head() ;
    while ( currentNode != null ) {
        if ( currentNode.element().equals(anElement) ) {
            return  true ;
        }
        currentNode = currentNode.next() ;
    }
    return  false ;
}
```

LinkedBag: frequencyOf()

```
// 상태 알아보기
.....
public int  frequencyOf (E anElement)
{
    int  frequencyCount = 0 ;
    LinkedNode<E>  currentNode = this.head() ;
    while ( currentNode != null ) {
        if ( currentNode.element().equals(anElement) ) {
            frequencyCount ++ ;
        }
        currentNode = currentNode.next() ;
    }
    return  frequencyCount ;
}
```

LinkedBag: any()

```
// 아무 원소 하나 얻으면 된다.
// 그렇다면, 어느 노드를 얻으면 효율적일까?
//
public E   any ()
{
    if ( this.isEmpty() ) {
        return   null ;
    }
    else {
        return   this.head().element() ;
    }
}
```

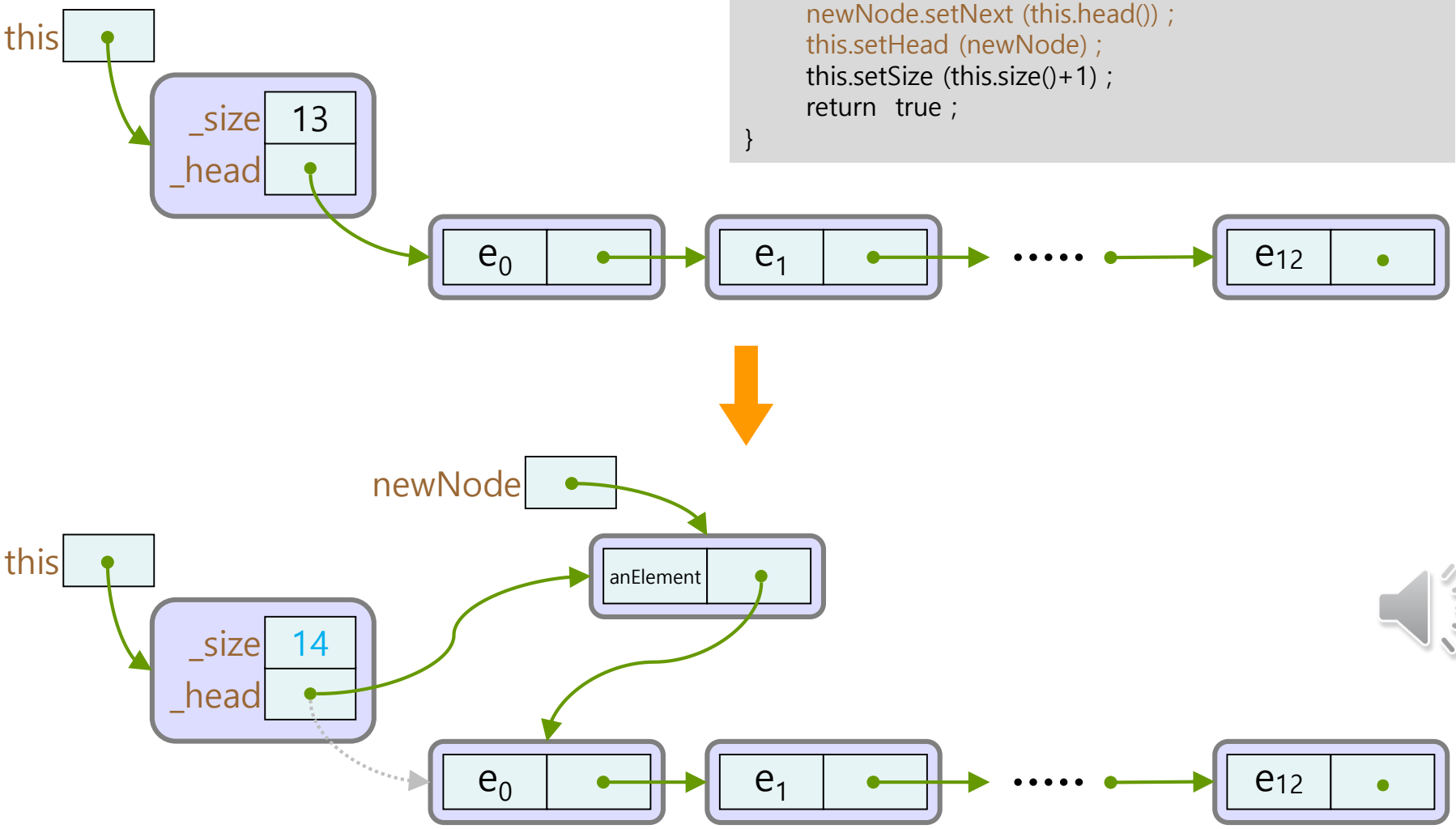
LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return  false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```

LinkedBag: add()

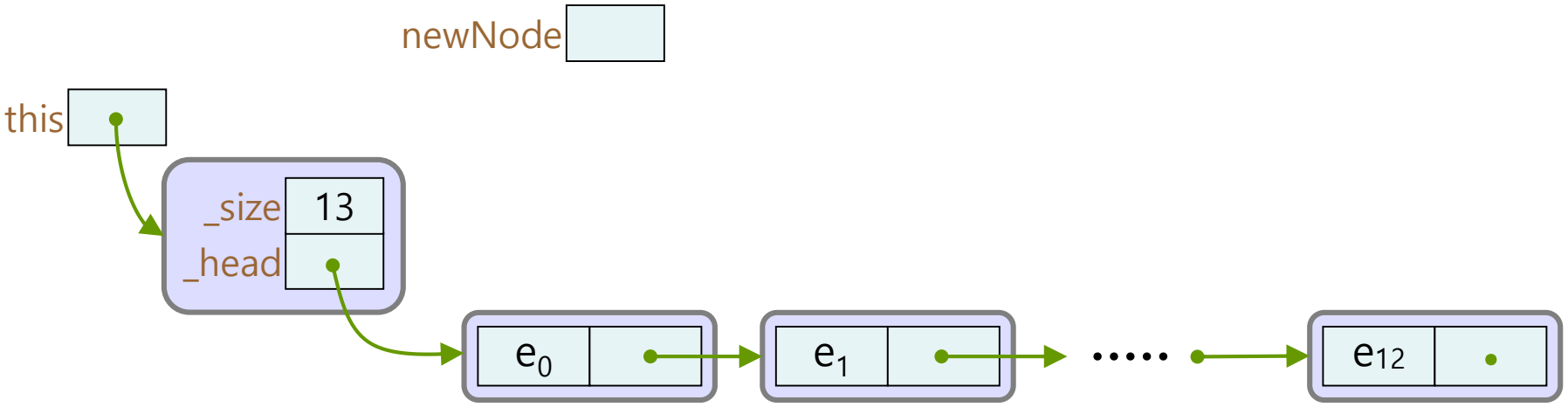
- 삽입 위치는?

```
else {
        LinkedNode<E>   newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return   true ;
}
```



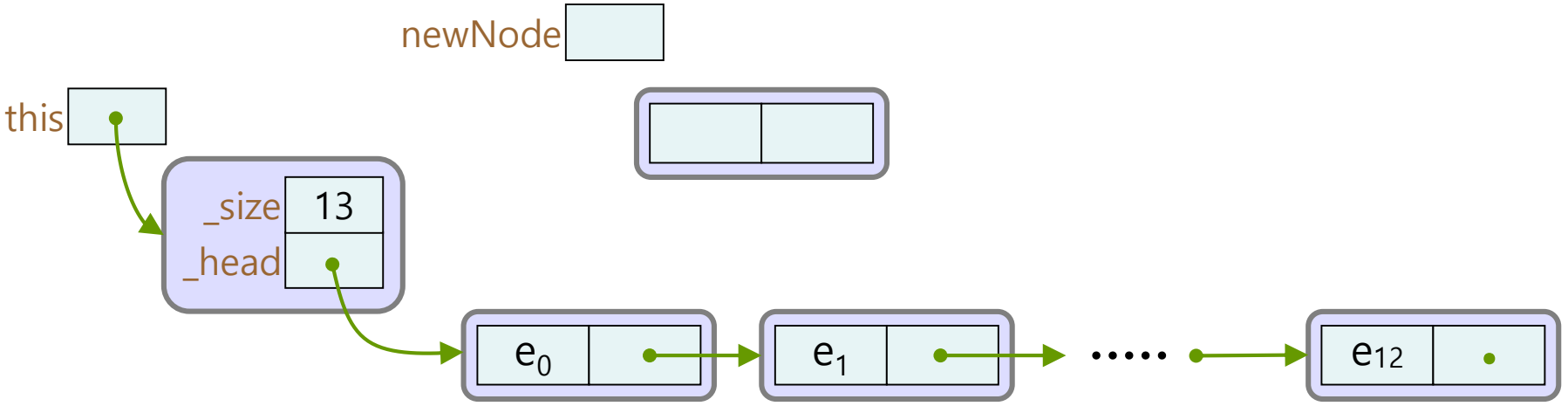
LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return  false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```



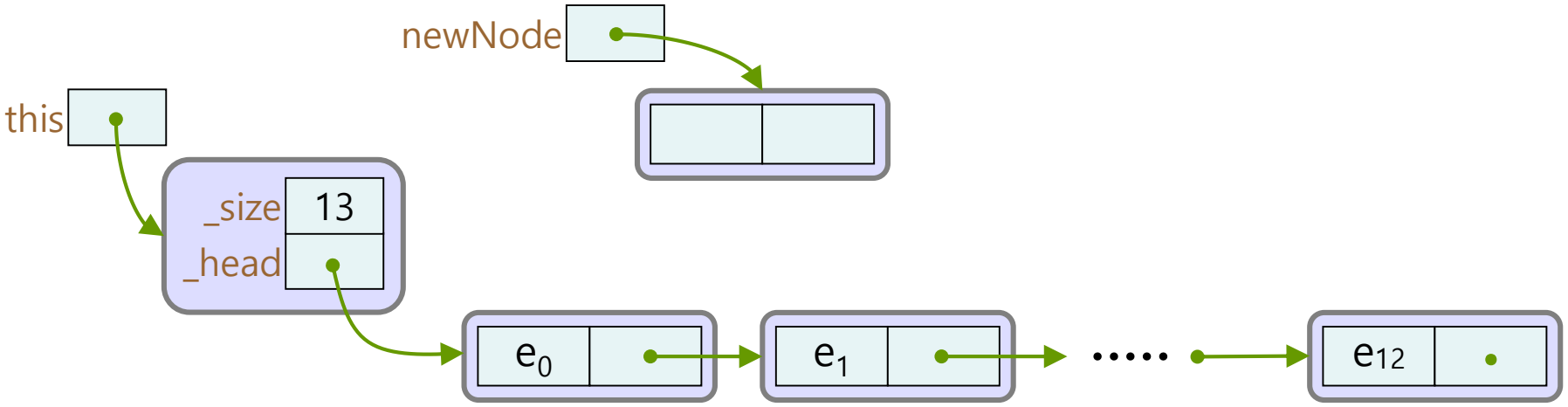
LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```



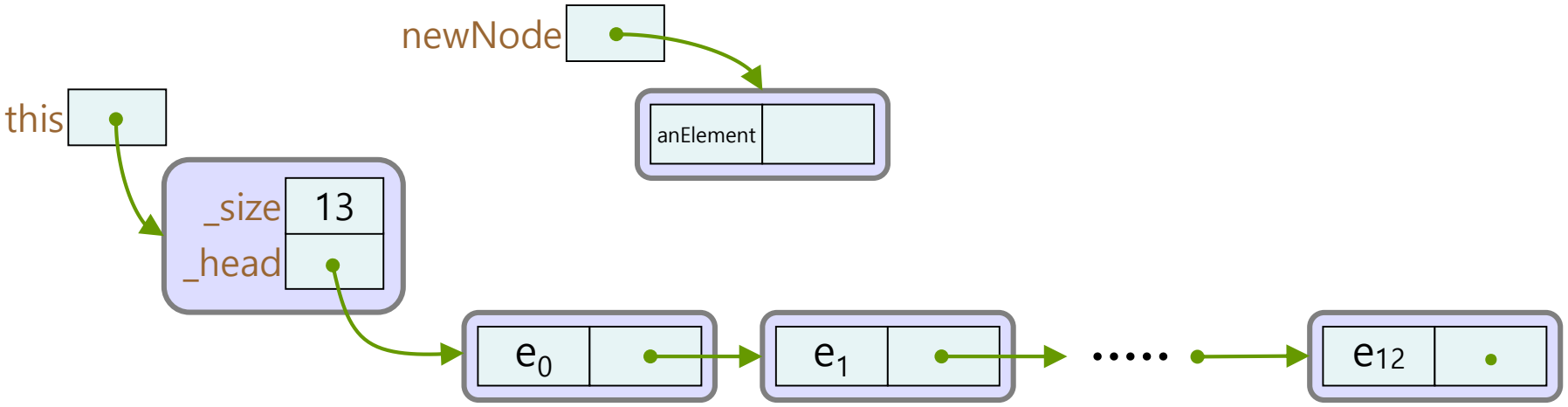
LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return  false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```



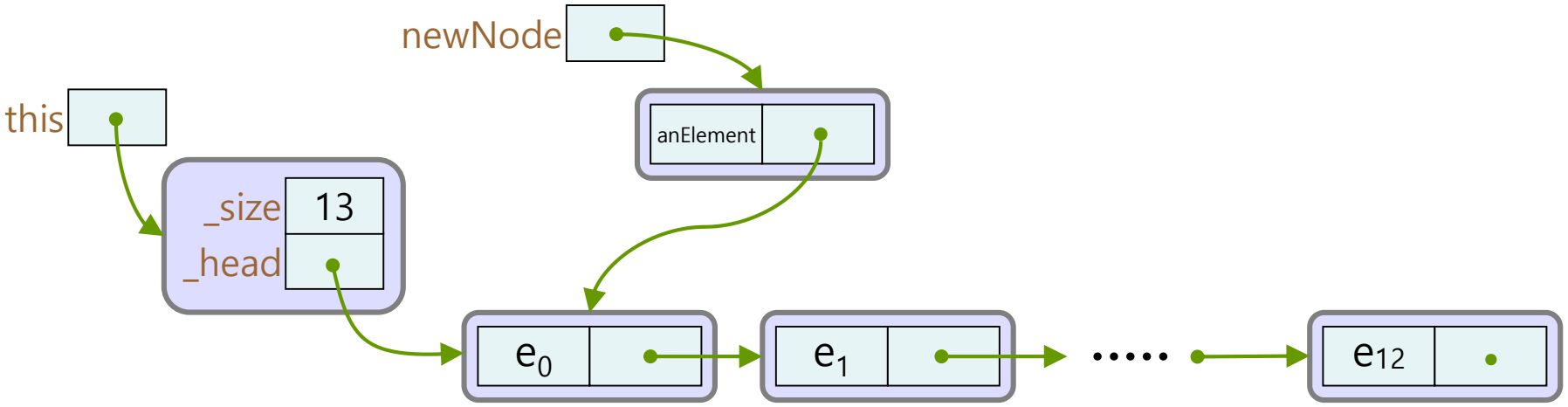
LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return  false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```



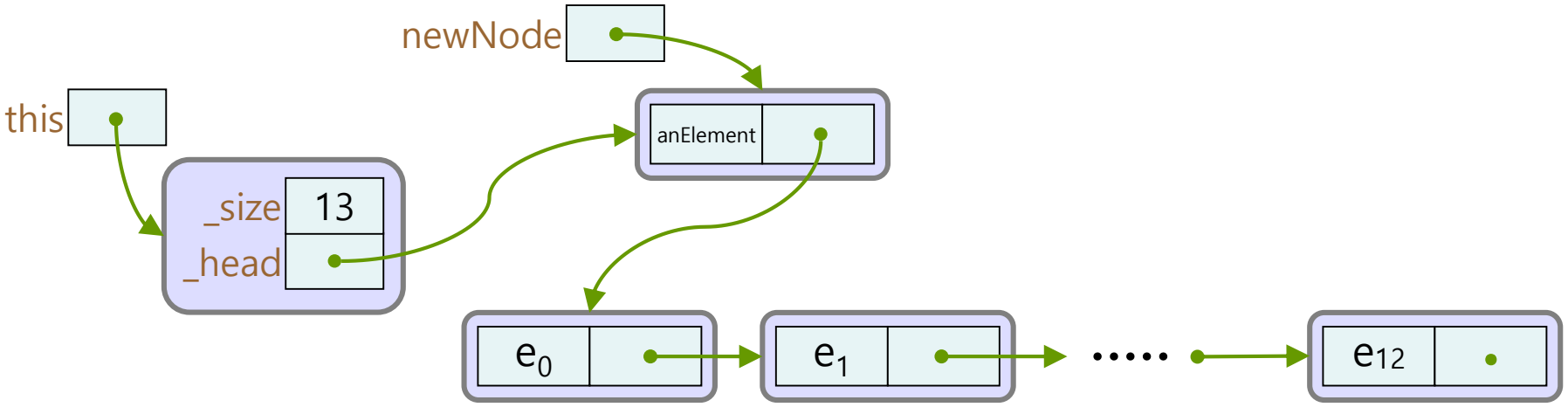
LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```



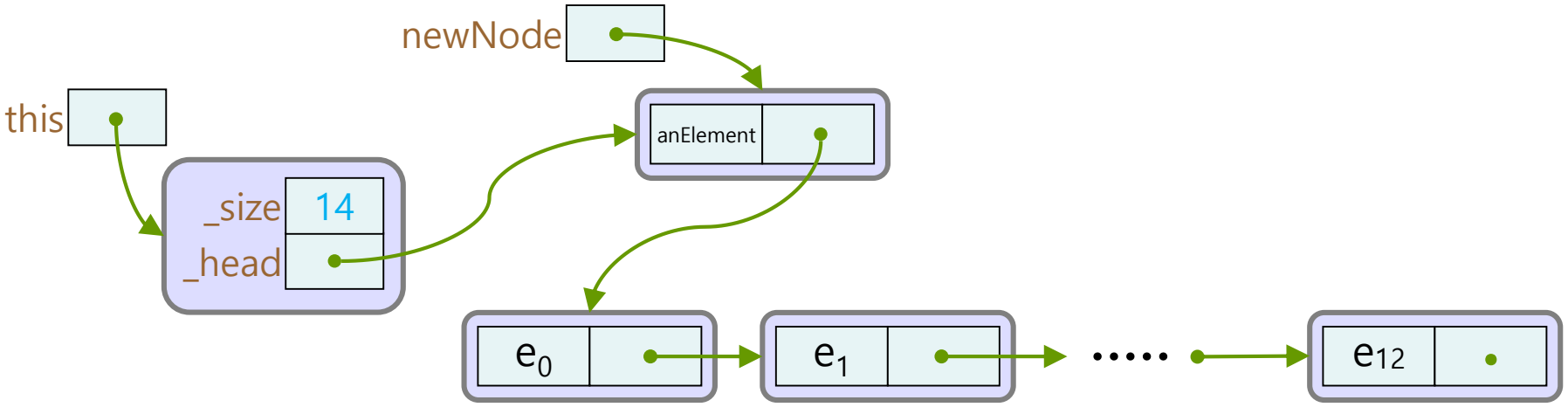
LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```



LinkedBag: add()

```
// 내용 바꾸기
public boolean  add (E anElement)
{
    if ( this.isFull() ) {
        return false ;
    }
    else {
        LinkedNode<E>  newNode = new LinkedNode<E>() ;
        newNode.setElement(anElement) ;
        newNode.setNext (this.head()) ;
        this.setHead (newNode) ;
        this.setSize (this.size()+1) ;
        return  true ;
    }
}
```



End of

"Linked Bag (1)"