



Жагсаалт. Холбоост суурилсан дүрслэл



- Жагсаалтын элемент ойд зоргоороо байрлана
- Элементээс элементед шилжихдээ нэмэлт мэдээлэл (**холбоос**) ашиглана

Ойн байршил

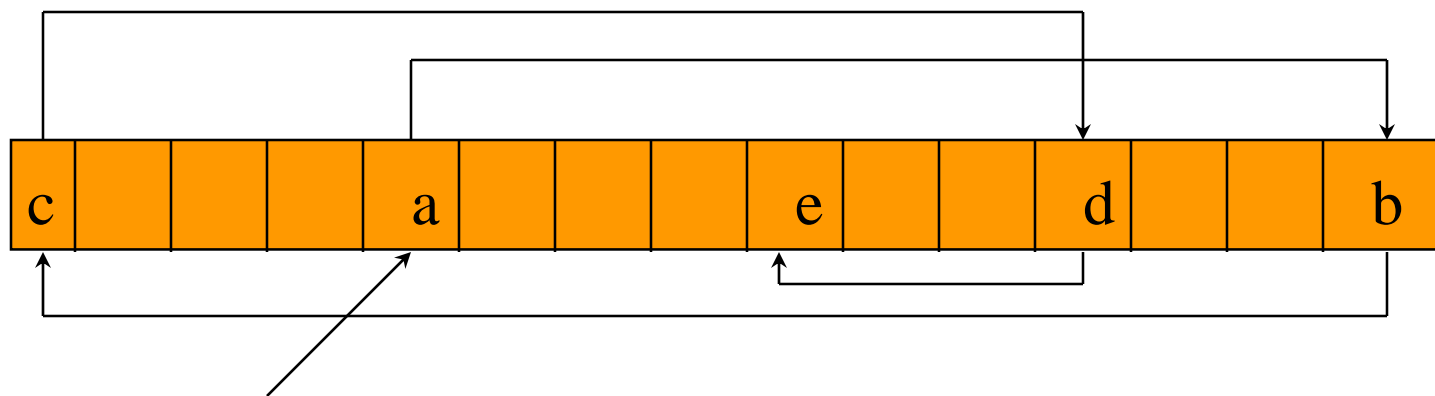
$L = (a, b, c, d, e)$ жагсаалт массив дүрслэлээр.

a	b	c	d	e										
---	---	---	---	---	--	--	--	--	--	--	--	--	--	--

Холбоост дүрслэлд зоргоороо/санамсаргүй байршина.

c				a				e			d			b
---	--	--	--	---	--	--	--	---	--	--	---	--	--	---

Холбоост дүрслэл

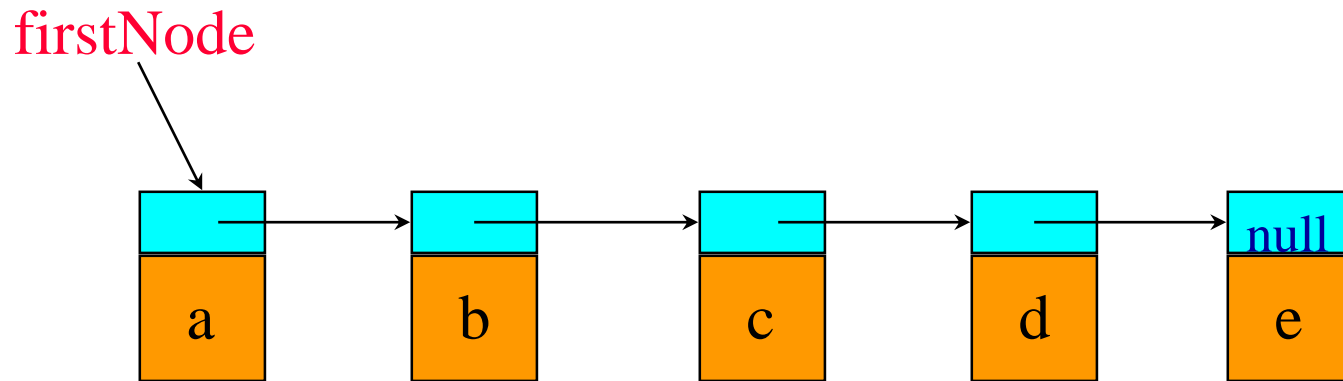


firstNode

е –ийн заагч (холбоос) null

хувьсагч firstNode эхний элемент a
–д хүрэхэд ашиглагдана

Холбоост жагсаалтыг зурах арга

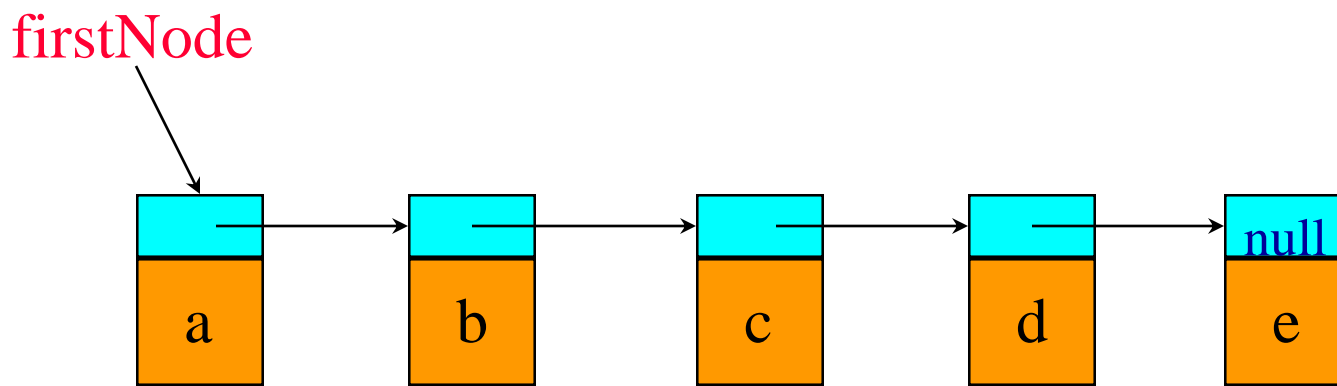


Зангилааны холбоос буюу заагчийн талбар



Зангилааны өгөгдлийн талбар

Гинж



- Гинж бол зангилаа бүр нэг элементийг илэрхийлэх холбоост жагсаалт.
- Энд холбоос буюу заагч нэг элементээс нөгөөд хүрэхийг зааж байна.
- Сүүлийн зангилааний заагч **null** байна.

Зангилааг дүрслэх

```
package dataStructures;
```

```
class ChainNode
```

```
{
```

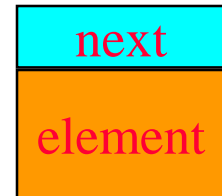
```
    // багцын харагдах өгөгдөл гишүүд
```

```
    Object element;
```

```
    ChainNode next;
```

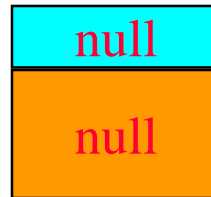
```
    // байгуулагчид
```

```
}
```

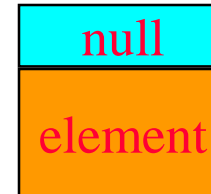


ChainNode –ийн байгуулагчид

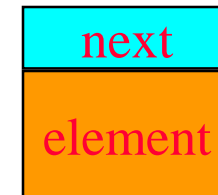
ChainNode() {}



ChainNode(Object element)
{ **this**.element = element; }

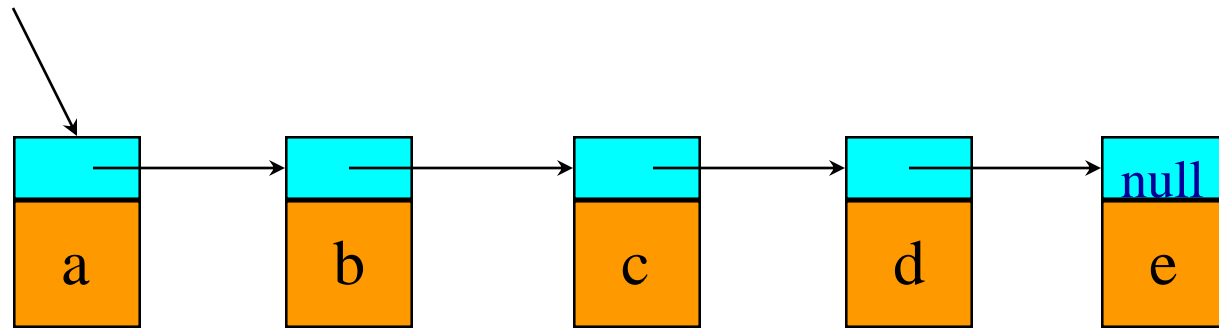


ChainNode(Object element, ChainNode next)
{ **this**.element = element;
 this.next = next; }



get(0)

firstNode



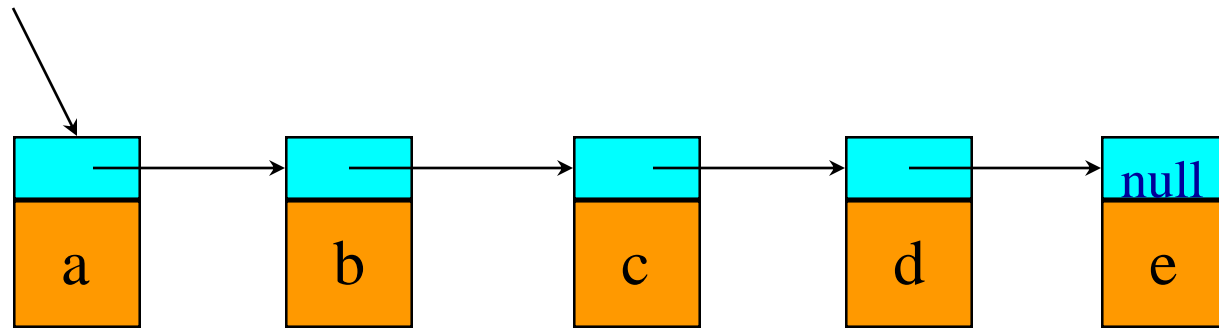
```
checkIndex(0);
```

```
desiredNode = firstNode; // ЭХНИЙ зангилааг авах
```

```
return desiredNode.element;
```


get(1)

firstNode



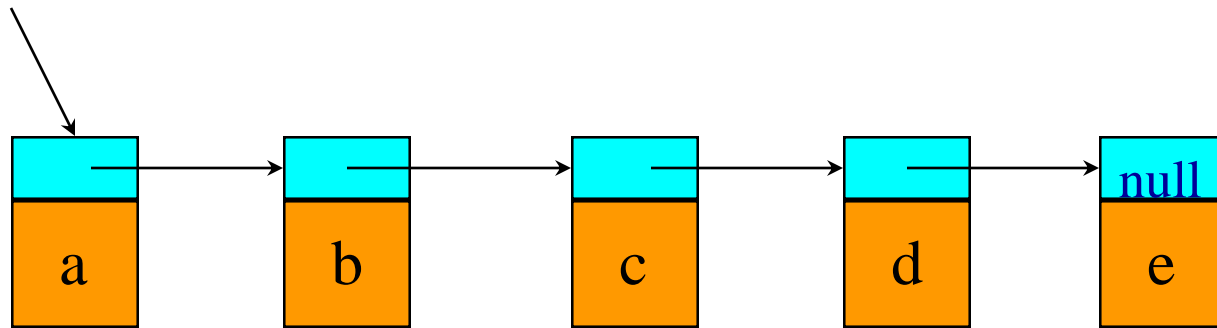
```
checkIndex(1);
```

```
desiredNode = firstNode.next; // хоёр дахь зангилааг авах
```

```
return desiredNode.element;
```

get(2)

firstNode



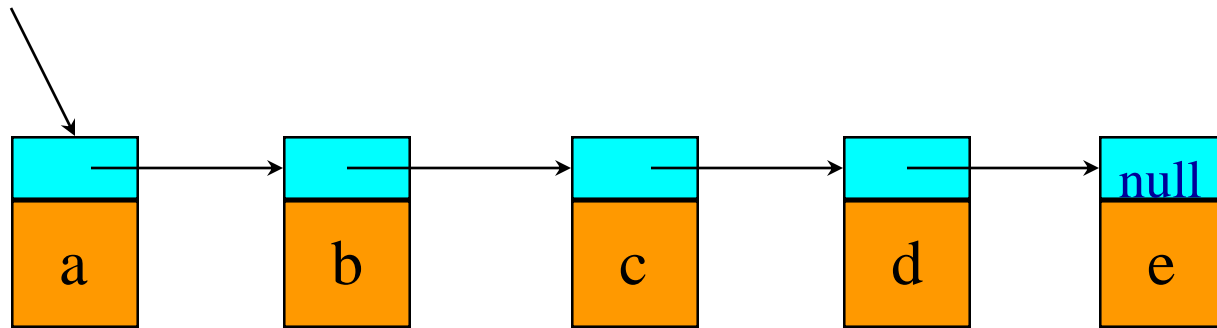
```
checkIndex(2);
```

```
desiredNode = firstNode.next.next; // гурав дахь зангилааг авах
```

```
return desiredNode.element;
```

get(5)

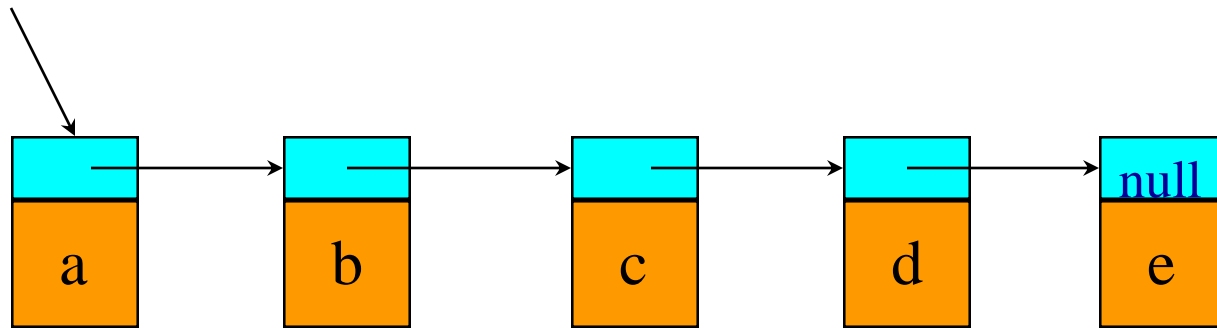
firstNode



```
checkIndex(5);           // онцгой тохиолдол унана  
desiredNode = firstNode.next.next.next.next.next;  
                        // desiredNode = null  
return desiredNode.element; // null.element
```

NullPointerException

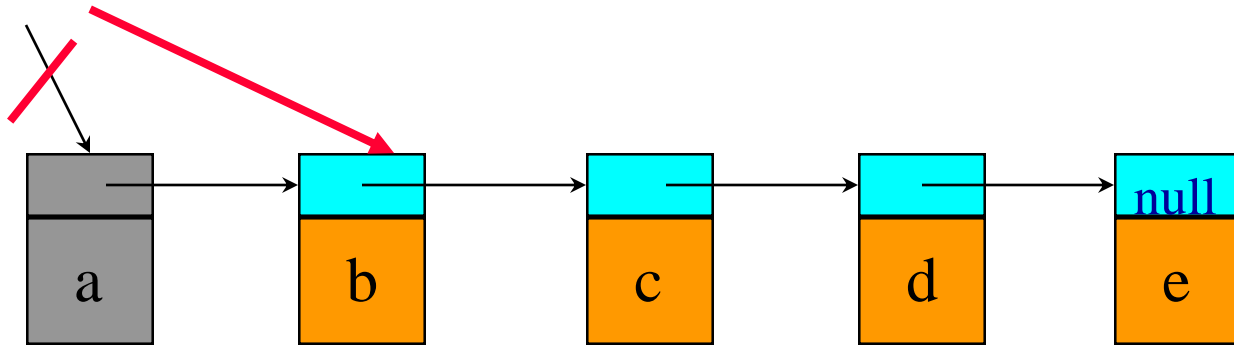
firstNode



```
desiredNode =  
    firstNode.next.next.next.next.next.next;  
    // компьютер тэнэг байдалд орж  
    // NullPointerException унана
```

Элемент устгах

firstNode

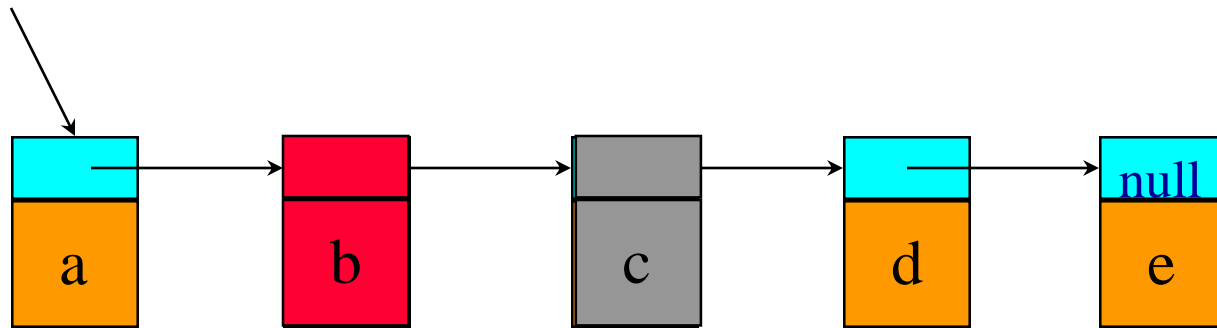


remove(0)

`firstNode = firstNode.next;`

remove(2)

firstNode

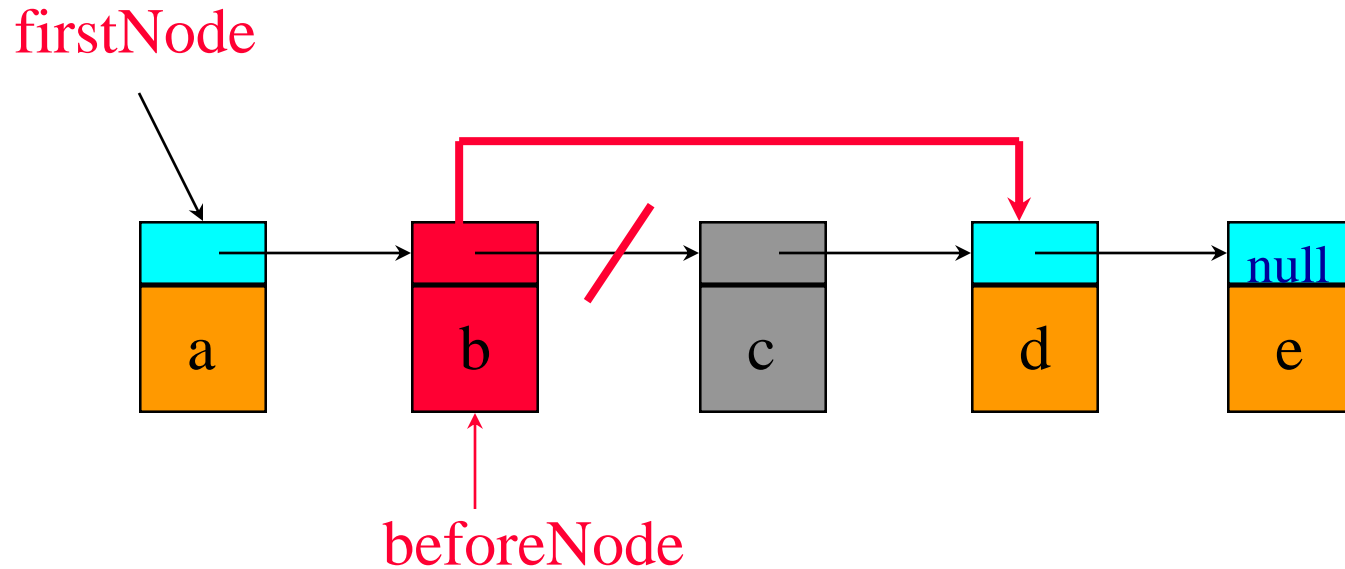


beforeNode

Зангилааг устгахаас өмнө түүнд хүрэх ёстой

`beforeNode = firstNode.next;`

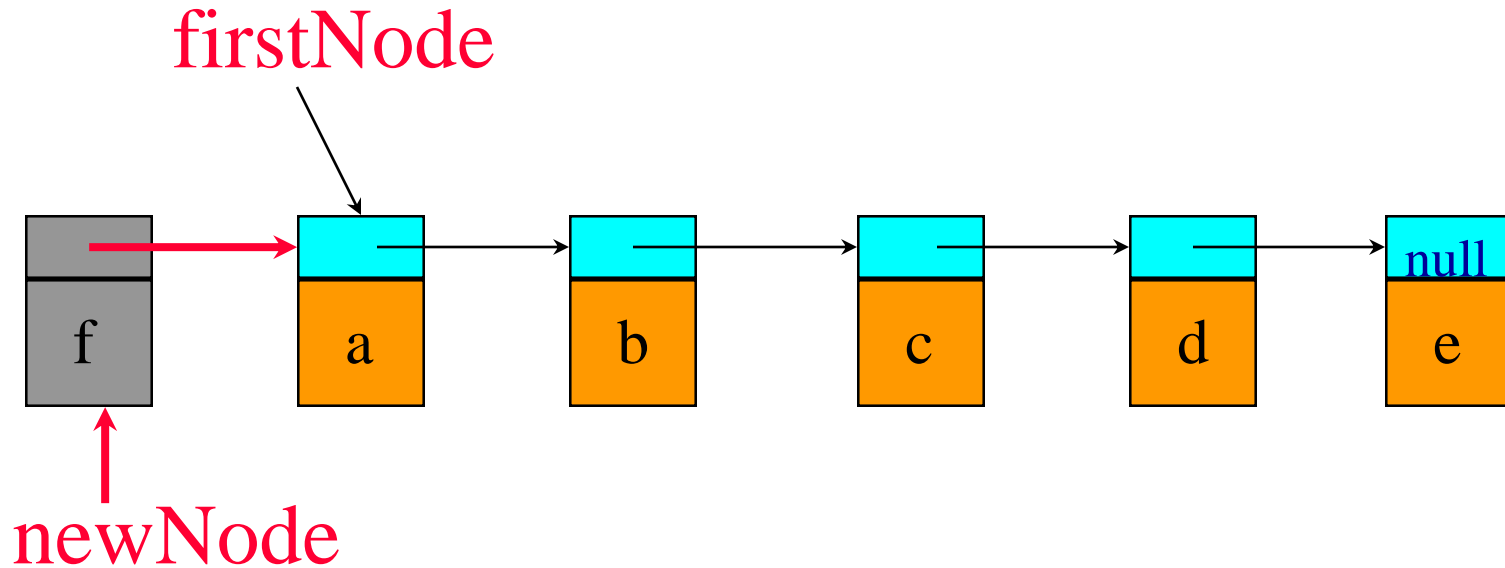
remove(2)



Одоо **beforeNode** –д байгаа заагчийг өөрчилж болно

`beforeNode.next = beforeNode.next.next;`

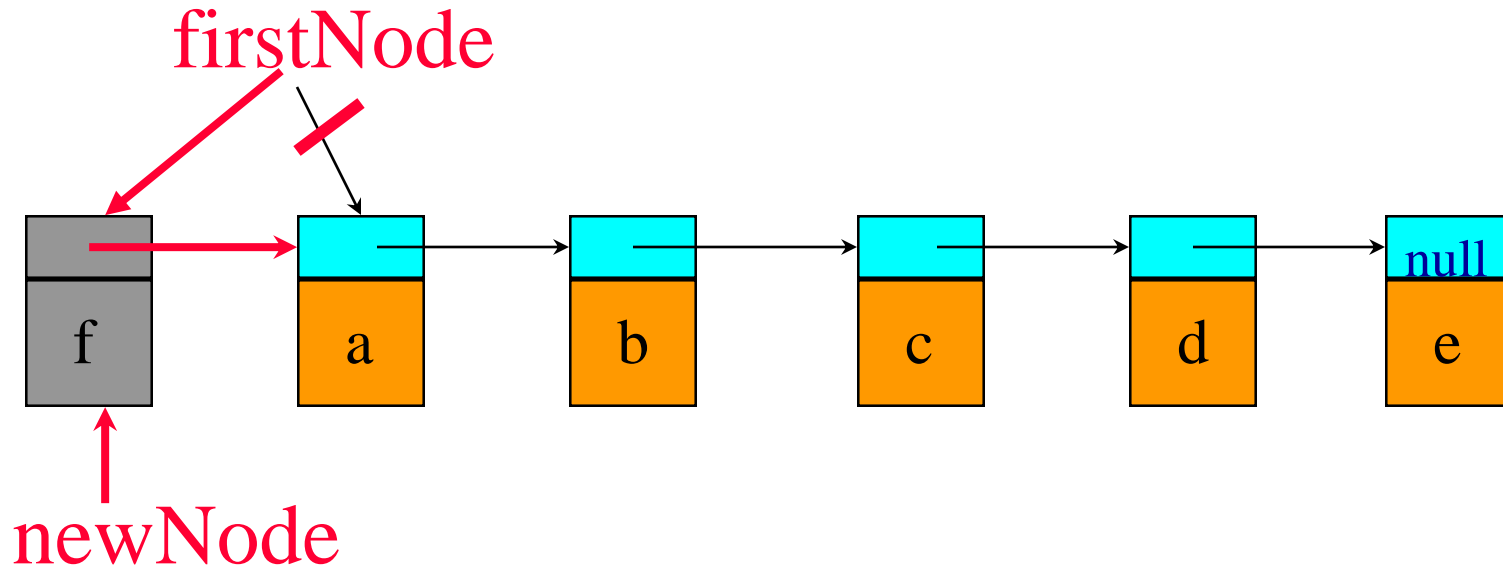
add(0, 'f')



Алхам 1: зангилааг үүсгэн өгөгдөл, холбоосын талбарыг тогтооно

```
ChainNode newNode =  
    new ChainNode(new Character('f'), firstNode);
```

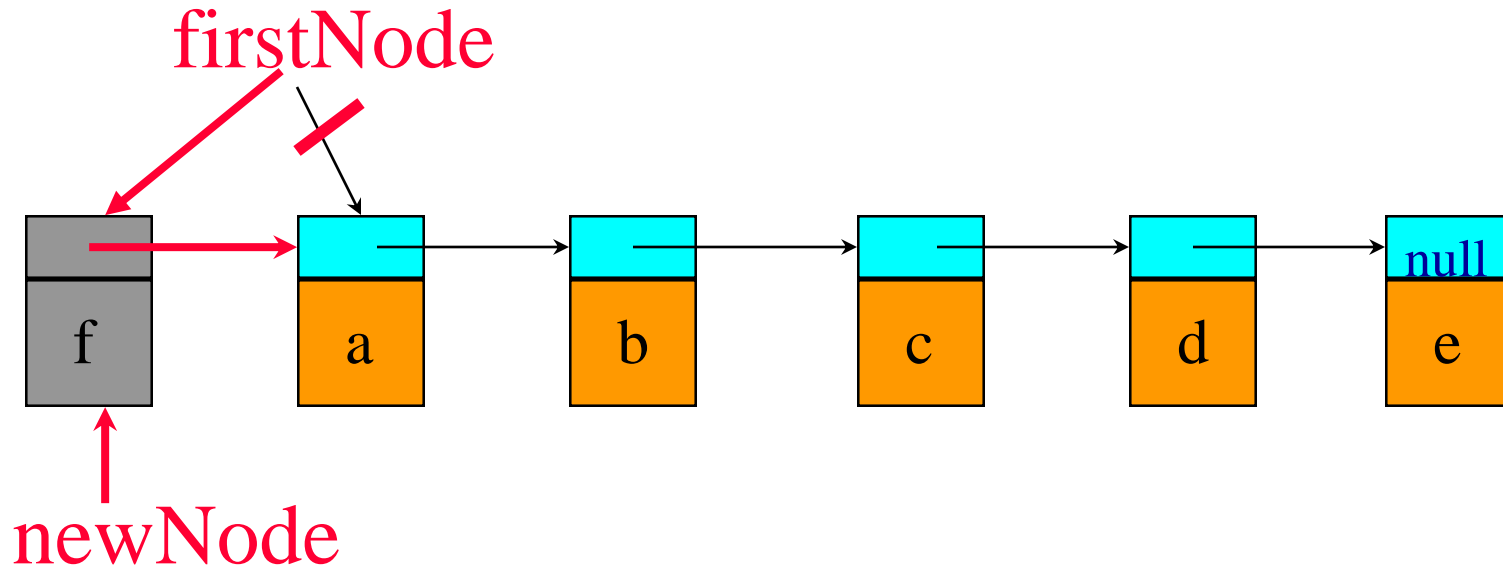

add(0, 'f')



Алхам 2: firstNode —Г ШИНЭЧЛЭНЭ

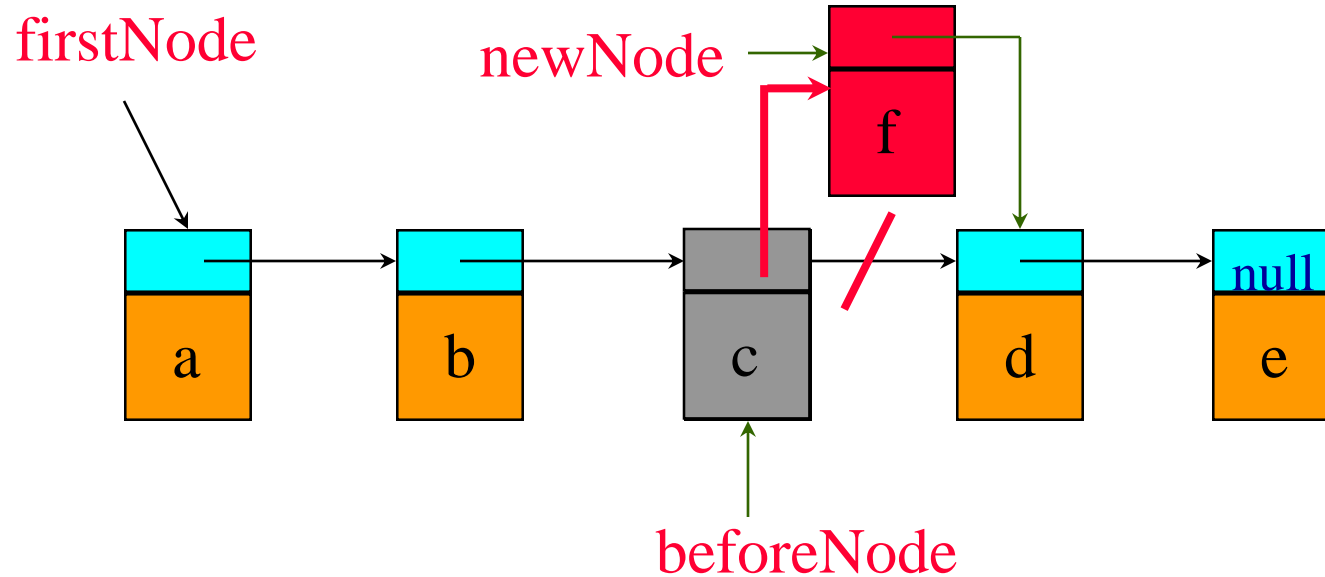
firstNode = newNode;

Нэг алхмын add(0,'f')



```
firstNode = new ChainNode(  
    new Character('f'), firstNode);
```

add(3, 'f')

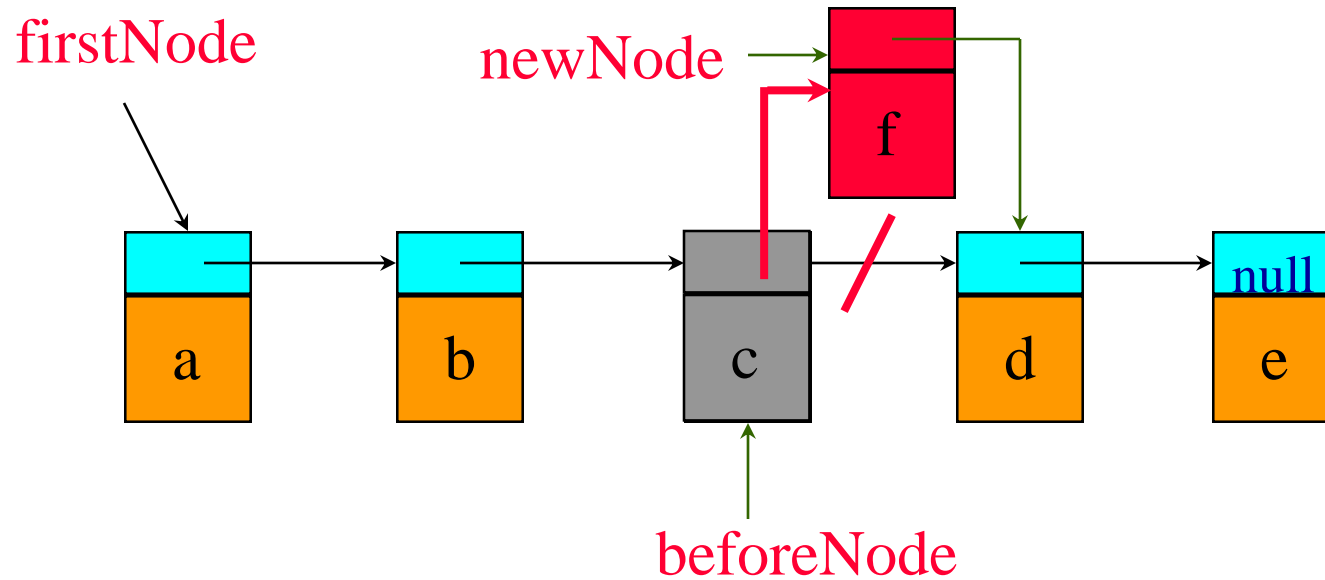


- эхлээд 2 гэсэн индекстэй зангилааг олох
- дараа нь зангилааг үүсгэн өгөгдөл, холбоосын талбарыг ТОГТООНО

```
ChainNode newNode = new ChainNode(new Character('f'),  
                                   beforeNode.next);
```

- ЭЦЭСТ НЬ beforeNode -г newNode –тай холбоно
- ```
beforeNode.next = newNode;
```

# Хоёр алхмын add(3,'f')



```
beforeNode = firstNode.next.next;
beforeNode.next = new ChainNode(new Character('f'),
 beforeNode.next);
```

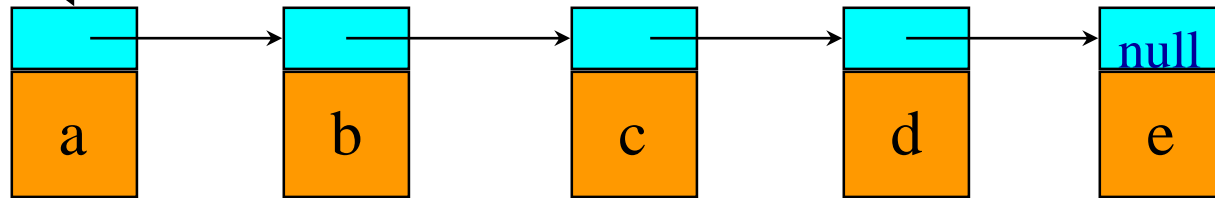


# Класс Chain



# Класс Chain

firstNode

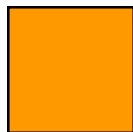


size = элементийн тоо

ChainNode —Г ашиглах



next (datatype ChainNode)



element (datatype Object)



# Класс Chain



```
/** LinearList холбоост хэрэгжүүлэлт*/
package dataStructures;
import java.util.*; // Iterator –г ашиглана
public class Chain implements LinearList
{
 // ӨГӨГДӨЛ гишүүд
 protected ChainNode firstNode;
 protected int size;

 // Chain –ий аргууд
}
```



## Байгуулагчид



*/\*\* хоосон жагсаалт үүсгэх \*/*

**public** Chain(**int** initialCapacity)

{

*// firstNode болон size –ий анхны утга*

*// null болон 0 байх болно*

}

**public** Chain()

{ **this**(0); }



# isEmpty арга



```
/** @return жагсаалт хоосон бол true */
public boolean isEmpty()
{return size == 0;}
```

# size() арга

/\*\* @return элементийн тоо \*/

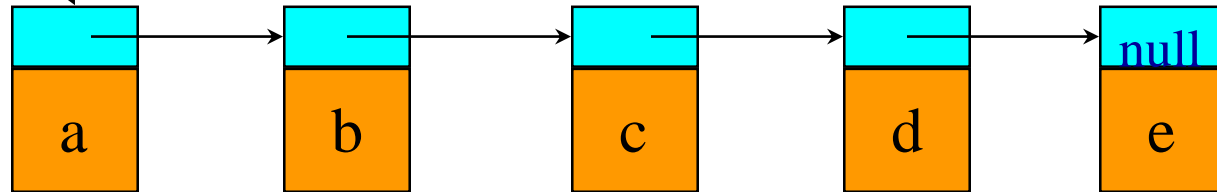
```
public int size()
{return size;}
```

## checkIndex арга

```
/** @throws index –н утга 0, size – 1 –н хооронд биш
 бол IndexOutOfBoundsException унана*/
void checkIndex(int index)
{
 if (index < 0 || index >= size)
 throw new IndexOutOfBoundsException
 ("index = " + index + " size = " + size);
}
```

firstNode

get арга



```
public Object get(int index)
```

```
{
```

```
 checkIndex(index);
```

```
 // хэрэгтэй зангилаанд хүрэх
```

```
 ChainNode currentNode = firstNode;
```

```
 for (int i = 0; i < index; i++)
```

```
 currentNode = currentNode.next;
```

```
 return currentNode.element;
```

```
}
```

# indexOf арга

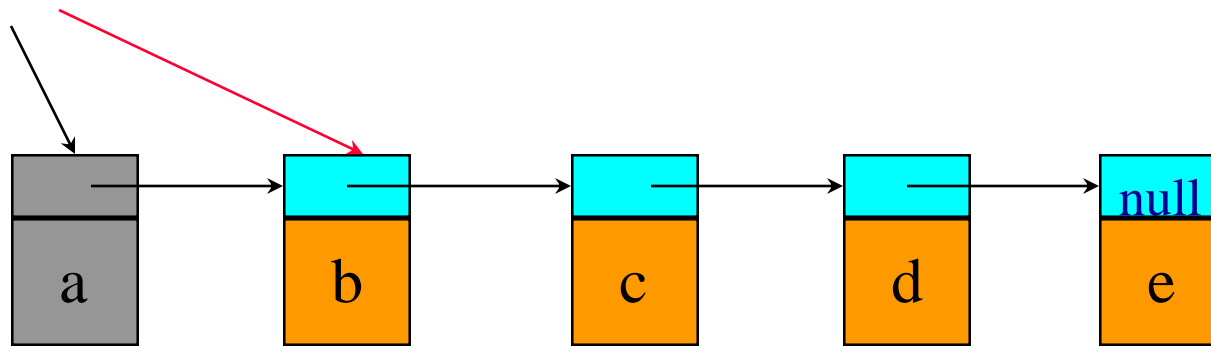
```
public int indexOf(Object theElement)
{
 // theElement –г хайх
 ChainNode currentNode = firstNode;
 int index = 0; // currentNode –н индекс
 while (currentNode != null &&
 !currentNode.element.equals(theElement))
 {
 // дараагийн зангилаанд хүрэх
 currentNode = currentNode.next;
 index++;
 }
}
```

# indexOf арга

```
// элемент олдсон уу?
if (currentNode == null)
 return -1;
else
 return index;
}
```

# Элементийг устгах

firstNode



`remove(0)`

`firstNode = firstNode.next;`



## Элементийг устгах



```
public Object remove(int index)
{
 checkIndex(index);

 Object removedElement;
 if (index == 0) // эхний зангилааг устгах
 {
 removedElement = firstNode.element;
 firstNode = firstNode.next;
 }
}
```

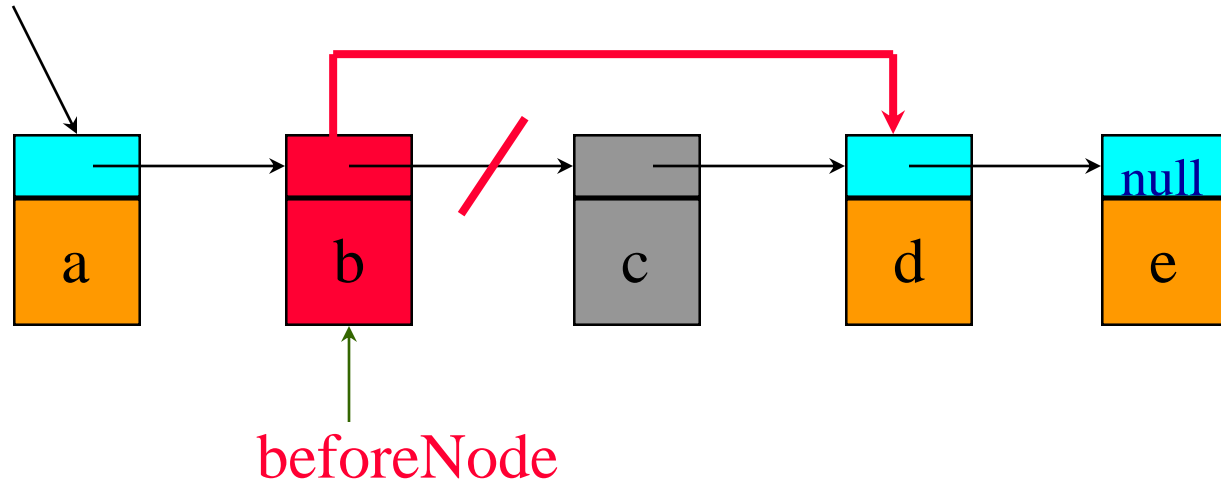




remove(2)



firstNode



**beforeNode** —г олж, заагчийг өөрчилнө.

**beforeNode.next = beforeNode.next.next;**



## Элементийг устгах

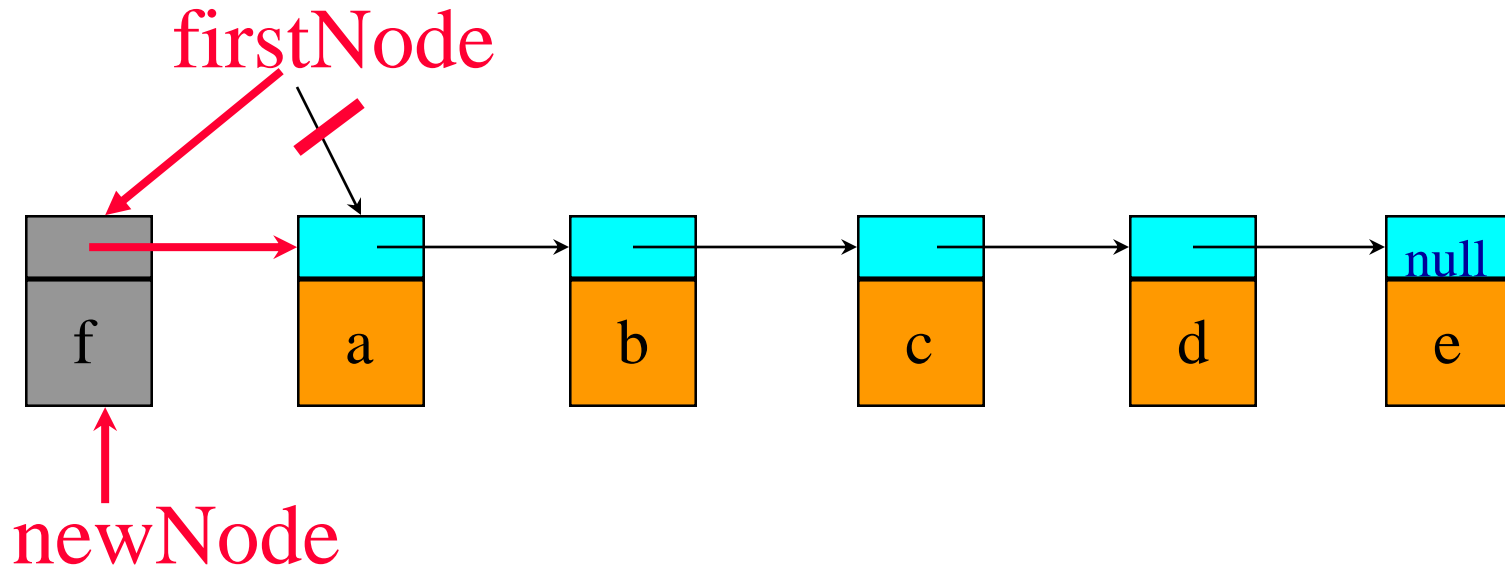


else

```
{ // хэрэгтэй зангилааны өмнөхыг олоход q –г ашиглана
 ChainNode q = firstNode;
 for (int i = 0; i < index - 1; i++)
 q = q.next;

 removedElement = q.next.element;
 q.next = q.next.next; // хайсан зангилаагаа устгах
}
size--;
return removedElement;
}
```

# Нэг алхмын add(0,'f')



```
firstNode = new ChainNode('f', firstNode);
```



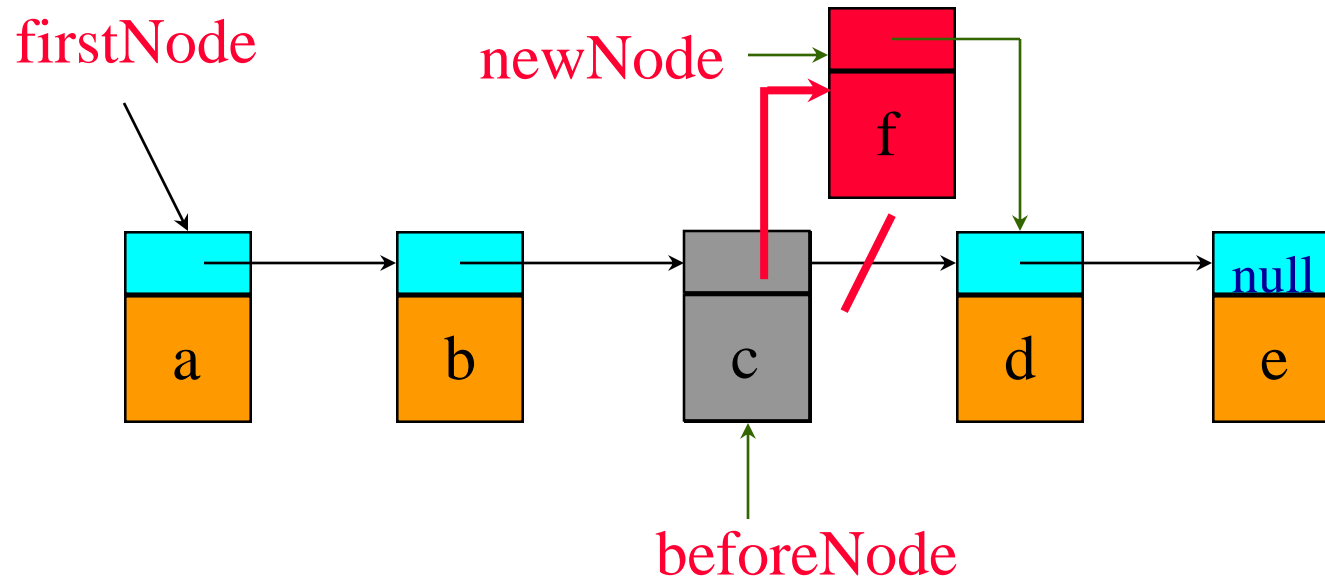
## ЭЛЕМЕНТ НЭМЭХ



```
public void add(int index, Object theElement)
{
 if (index < 0 || index > size)
 // индекс буруу байна
 throw new IndexOutOfBoundsException
 ("index = " + index + " size = " + size);

 if (index == 0)
 // эхэнд оруулах
 firstNode = new ChainNode(theElement, firstNode);
```

# Хоёр алхмын add(3,'f')



```
beforeNode = firstNode.next.next;
beforeNode.next = new ChainNode('f', beforeNode.next);
```

## Элемент нэмэх

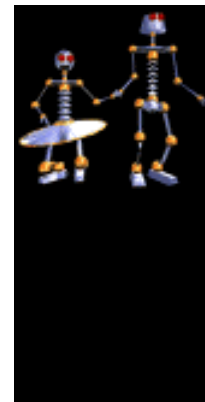
else

```
{ // шинэ элементийн өмнөхийг олох
 ChainNode p = firstNode;
 for (int i = 0; i < index - 1; i++)
 p = p.next;

 // p –ийн дараа оруулах
 p.next = new ChainNode(theElement, p.next);
}
size++;
}
```

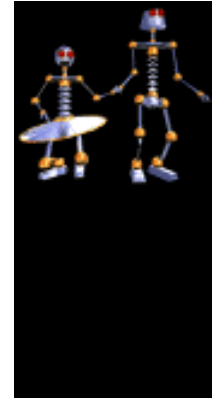
# Чанарын үзүүлэлт

Үйлдэл бүрийг 40,000 давтлаа



# Чанарын үзүүлэлт

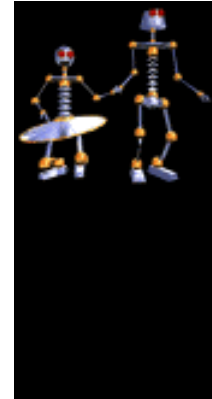
Үйлдэл бүрийг 40,000 давтлаа



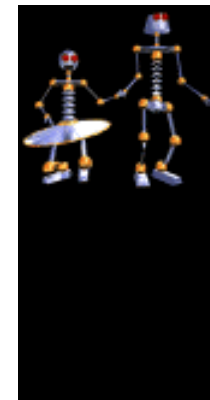
| Үйлдэл        | FastArray | LinkedList | Chain |
|---------------|-----------|------------|-------|
| авах          | 5.6ms     | 157sec     |       |
| нэмэх-сайн    | 31.2ms    | 304ms      |       |
| нэмэх-дундаж  | 5.8sec    | 115sec     |       |
| нэмэх-муу     | 11.8sec   | 157sec     |       |
| устгах-сайн   | 8.6ms     | 13.2ms     |       |
| устгах-дундаж | 5.8sec    | 149sec     |       |
| устгах-муу    | 11.7sec   | 157sec     |       |



# Чанарын үзүүлэлт



Индексжүүлсэн AVL ('62 - G.M **A**delson-**V**elsky & E.M **L**andis)  
мод (IAVL)



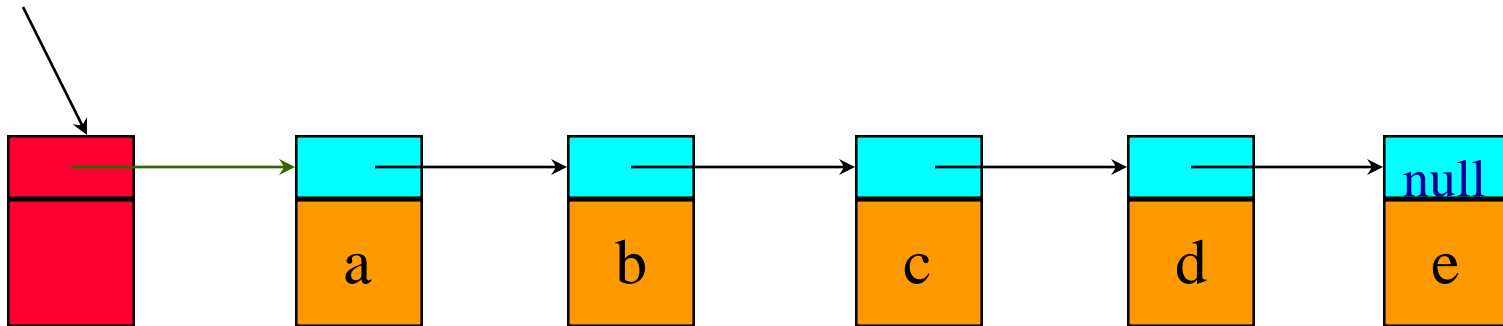
# Чанарын үзүүлэлт

Индексжүүлсэн AVL мод (IAVL)

| Үйлдэл        | FastArrayLinearList | Chain  | IAVL   |
|---------------|---------------------|--------|--------|
| авах          | 5.6ms               | 157sec | 63ms   |
| нэмэх-сайн    | 31.2ms              | 304ms  | 253ms  |
| нэмэх-дундаж  | 5.8sec              | 115sec | 392ms  |
| нэмэх-муу     | 11.8sec             | 157sec | 544ms  |
| устгах-сайн   | 8.6ms               | 13.2ms | 1.3sec |
| Устгах-дундаж | 5.8sec              | 149sec | 1.5sec |
| устгах-муу    | 11.7sec             | 157sec | 1.6sec |

# Толгойн зангилаатай гинж

headerNode





# Толгойн зангилаатай хоосон гинж

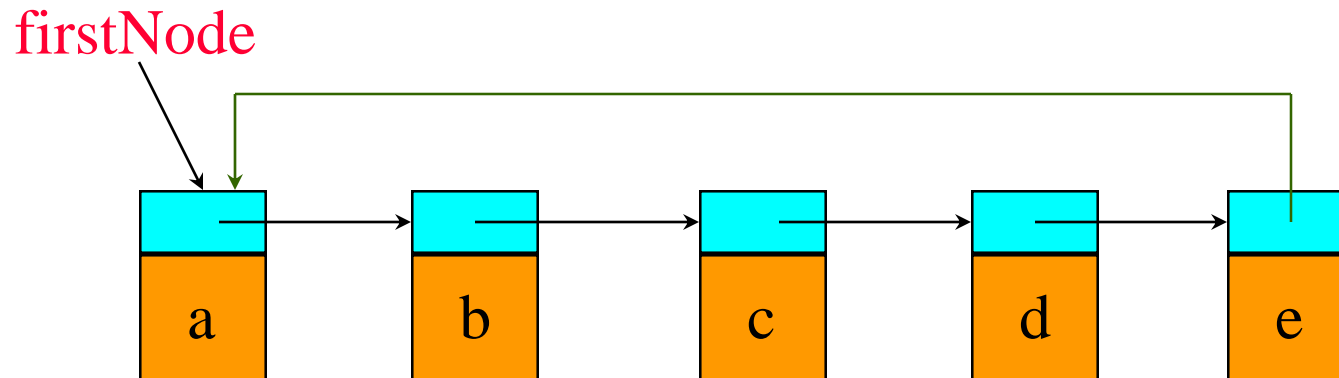


headerNode





# Цагириг жагсаалт





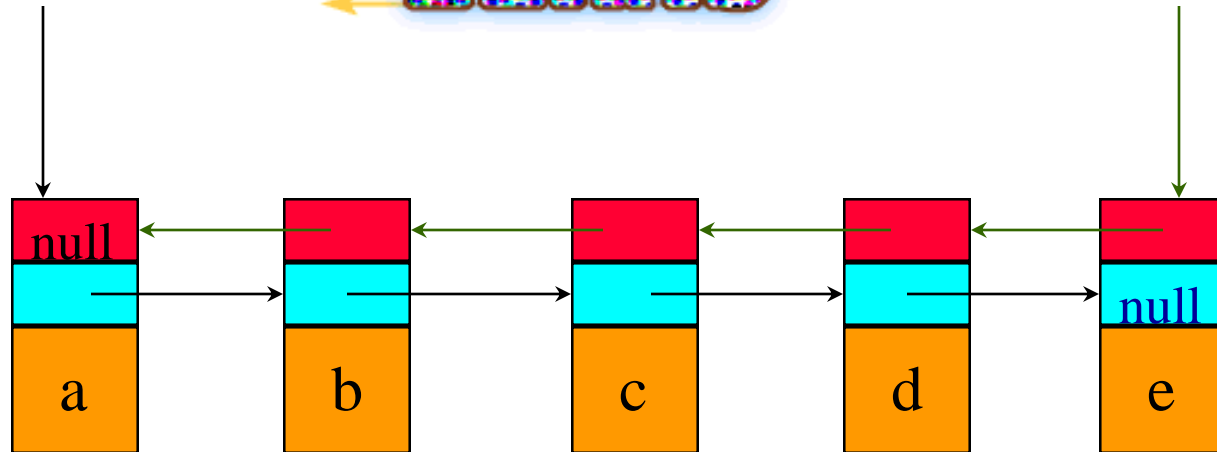
# Хос холбоост жагсаалт



firstNode



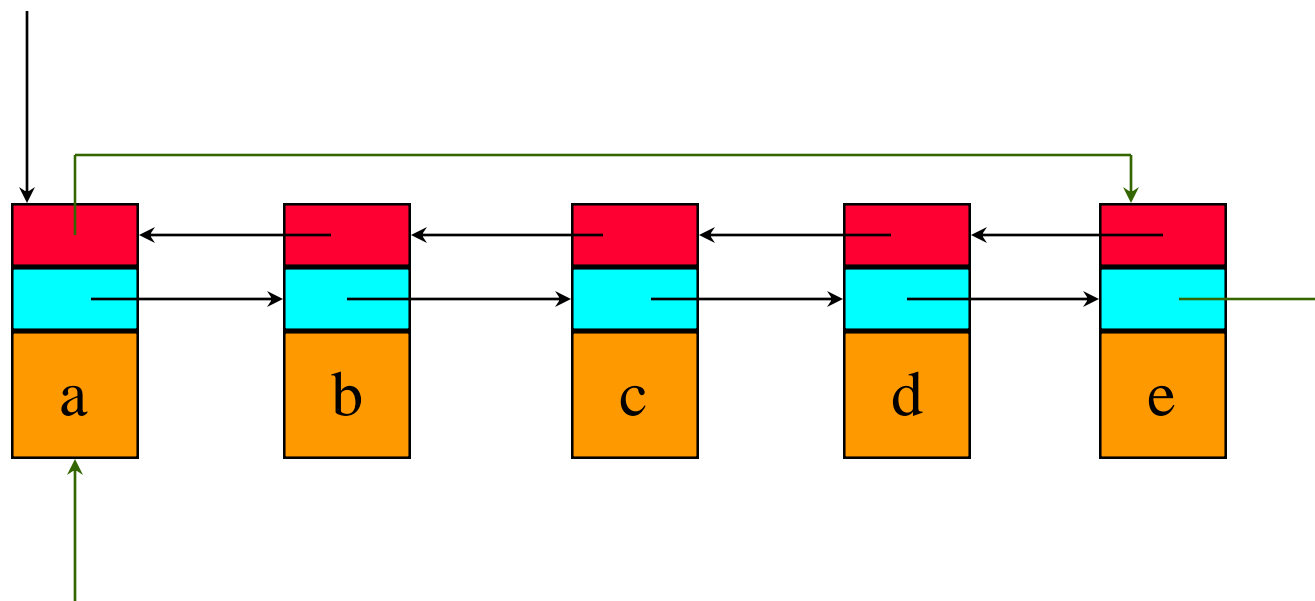
lastNode





# Хос холбоост цагариг жагсаалт

firstNode

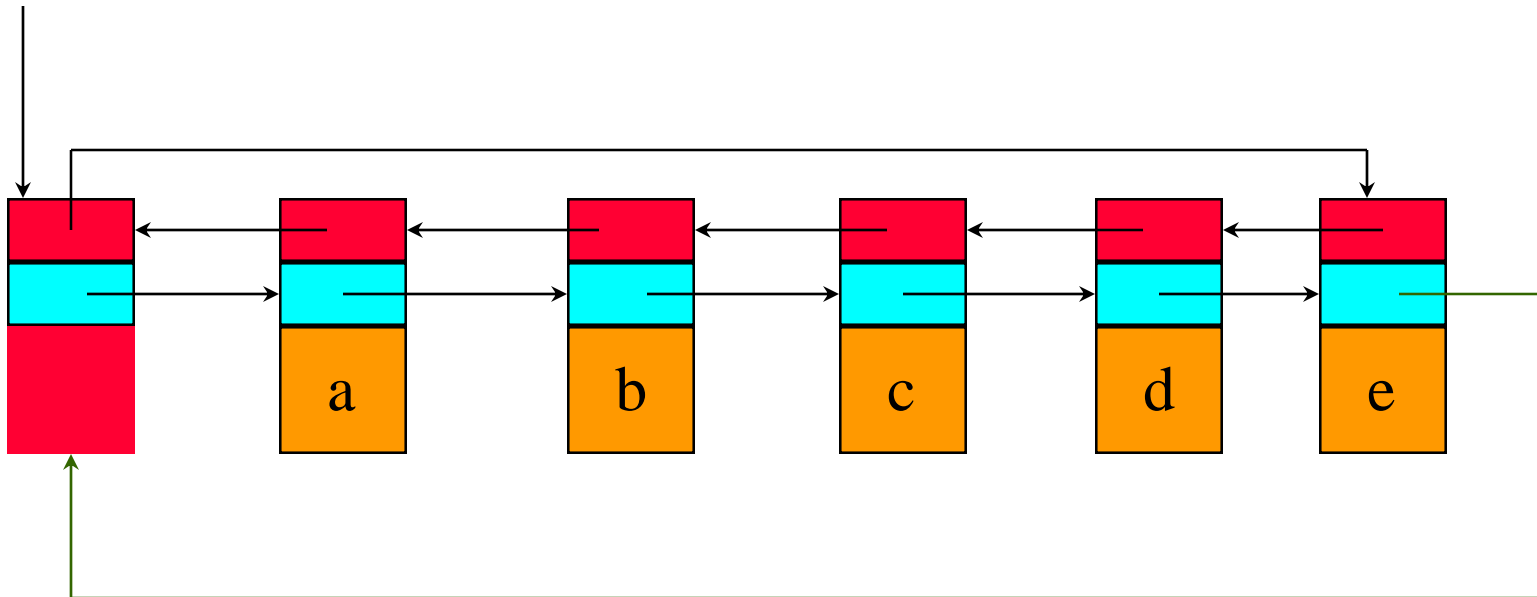




# Толгойн зангилаатай хос холбоост цагариг жагсаалт



headerNode

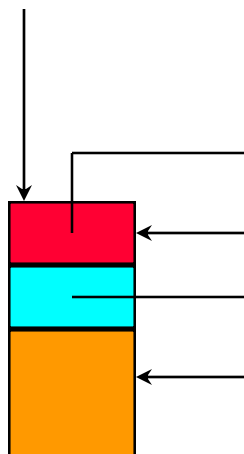




# Толгойн зангилаатай хос холбоост цагариг хоосон жагсаалт



headerNode





# java.util.LinkedList



- Шугаман жагсаалтын холбоост хэрэгжүүлэлт.
- Толгойн зангилаатай хос холбоост цагариг жагсаалт.
- **LinkedList** –ийн бүх аргаас гадна олон зүйл энд бий.