```
package cn.sxt.mycollection;
/**
  * 自定义一个链条
  * @author 江
  *
  */
public class SxtLinkedList<E> {
            private Node first;
            private Node last;

            private int size;


            public void checkRange(int index) {
               if(index<0||index>size) {
                        throw new RuntimeException("索引数字不合
法:"+index);
               }
            }
            public Object get(int index) {
               checkRange(index);


               Node temp=getNode(index);
               return temp.element;
            }

            public Node getNode(int index) {
               checkRange(index);
               Node temp;
```

```
    //效率增加，二分进行比较
    if(index<=size>>1) {
            temp=first;
            for(int i=0;i<index;i++) {
                    temp=temp.next;
            }
    }else {
            temp=last;
            for(int i=size-1;i>index;i--) {
                    temp=temp.previous;
            }
    }
    return temp;
}

//[]
//["a","b","c"]
public void add(E element) {
    Node node =new Node(element);

    if(first==null) {
            first=node;
            last=node;
            size++;
    }else {
            node.previous=last;
            node.next=null;

            last.next=node;
            last=node;
```

```
                size++;
        }
    }

    public void add(int index,E element) {
     Node newNode=new Node(element);
     Node temp=getNode(index);
     Node up=temp.previous;
     Node dowm=temp.next;
     Node f=first;
     Node l=last;

     checkRange(index);
          if(up!=null) {
               up.next=newNode;
               newNode.previous=up;
               newNode.next=temp;
               temp.previous=newNode;
     }

          if(index==0) {
                first=newNode;
                newNode.next=f;
                f.previous=newNode;
          }

          if(index==size) {
                 last=newNode;
                 newNode.previous=l;
                  l.next=newNode;
```

```
        }
     size++;
    }

    public void remove(int index) {
        checkRange(index);
        Node temp=getNode(index);
        Node up=temp.previous;
        Node down=temp.next;

        if(up!=null) {
                    up.next=down;
        }
        if(down!=null) {
                    down.previous=up;
        }

        //被删除的是第一个元素时
        if(index==0) {
                    first=down;
        }

        //被删除的是最后一个元素时
        if(index==size-1) {
                    last=up;
        }
        size--;

    }
```

```
    @Override
    public String toString() {
        StringBuilder sb=new StringBuilder();
        Node temp=first;
        sb.append("[");
        while(temp!=null) {
                sb.append(temp.element+",");
                temp=temp.next;
        }
        sb.setCharAt(sb.length()-1, ']');
        return sb.toString();
    }



    public static void main(String[] args) {
            SxtLinkedList<String> list=new SxtLinkedList<>
();

            list.add("a");
            list.add("b");
            list.add("c");
            list.add("d");
            list.add("e");
            list.add("f");

            System.out.println(list);
            System.out.println(list.get(5));

            list.remove(0);
```

```
                System.out.println(list);
            System.out.println(list.size);
                list.add(4,"ih");
                System.out.println(list);
        }
}
package cn.sxt.mycollection;
public class Node {

        Node previous;   //上一个节点
        Node next;       //下一个节点
        Object element;     //元素数据


        public Node(Node previous, Node next, Object element) {
                super();
                this.previous = previous;
                this.next = next;
                this.element = element;
        }
        public Node(Object element) {
                super();
                this.element = element;
        }



}
```