golang 之 linkedList

作者：xhaoxiong
原文链接：https://ld246.com/article/1538559474446
来源网站：链滴

```
package linkList

import (
    "github.com/pkg/errors"
    "github.com/spf13/cast"
)

type LinkList struct {
    node
    dummyHead *node //虚拟头节点
    size      int
}

type node struct {
    Elem interface{}
    Next *node
}

func main() {
    LinkList := &LinkList{dummyHead: &node{nil, nil}}
    for i := 0; i < 10; i++ {
        LinkList.AddFirst(i)
    }
    LinkList.ToString()
    LinkList.Set(10, 1)
    LinkList.ToString()
    LinkList.RemoveFirst()
    LinkList.ToString()
    LinkList.RemoveLast()
    LinkList.ToString()
}

func Init() *LinkList {
    return &LinkList{dummyHead: &node{nil, nil}}
}

func (this *LinkList) Remove(index int) (elem interface{}) {
    if index < 0 || index > this.size {
        panic("out of range")
    }
    prev := this.dummyHead

    for i := 0; i < index; i++ {
        prev = prev.Next
    }
    retNode := prev.Next
    prev.Next = retNode.Next
    retNode.Next = nil
    elem = retNode.Elem
    this.size--
    return
}
```

```
func (this *LinkList) RemoveFirst() {
    this.Remove(0)
}

func (this *LinkList) RemoveLast() {
    this.Remove(this.size - 1)
}

func (this *LinkList) IsEmpty() (bool) {
    return this.size == 0
}

func (this *LinkList) Add(elem interface{}, index int) {
    if index < 0 || index > this.size {
        panic(errors.New("out of range "))
    }

    prev := this.dummyHead
    //找到index处对应的前一个node节点
    for i := 0; i < index; i++ {
        prev = prev.Next
    }

    newNode := &node{Elem: elem}
    newNode.Next = prev.Next
    prev.Next = newNode
    this.size++

}

func (this *LinkList) AddFirst(elem interface{}) {
    this.Add(elem, 0)
}

func (this *LinkList) AddLast(elem interface{}) {
    this.Add(elem, this.size)
}

func (this *LinkList) Get(index int) (elem interface{}) {
    if index < 0 || index >= this.size {
        panic("out of range ")
    }

    cur := this.dummyHead.Next

    for i := 0; i < index; i++ {
        cur = cur.Next
    }

    return cur.Elem
}

func (this *LinkList) GetFirst() (elem interface{}) {
    return this.Get(0)
```

```
}

func (this *LinkList) GetLast() (elem interface{}) {
    return this.Get(this.size - 1)
}

func (this *LinkList) Set(elem interface{}, index int) {
    if index < 0 || index >= this.size {
        panic("out of range ")
    }

    cur := this.dummyHead.Next

    for i := 0; i < index; i++ {
        cur = cur.Next
    }

    cur.Elem = elem
}

func (this *LinkList) Contains(elem interface{}) (bool) {

    cur := this.dummyHead.Next
    for i := 0; i < this.size; i++ {
        cur = cur.Next
    }

    if cur.Elem == elem {
        return true
    }
    return false
}

func (this *LinkList) ToString() string {
    cur := this.dummyHead.Next
    s := cast.ToString(cur.Elem) + "->"
    for cur.Next != nil {
        cur = cur.Next
        s += cast.ToString(cur.Elem) + "->"

    }
    s += "nil"
    return s
}

func (this *LinkList) GetSize() int {
    return this.size
}
```