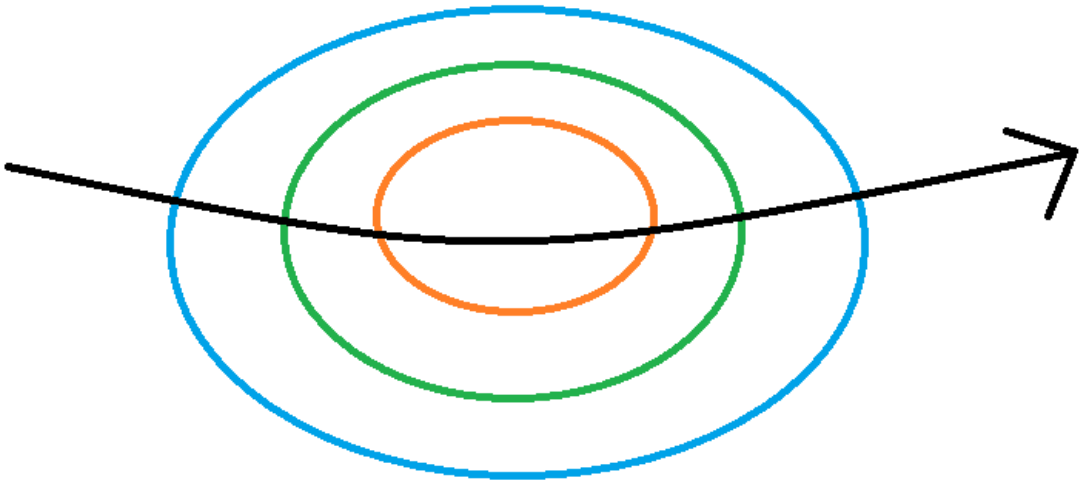
栈

什么是栈?

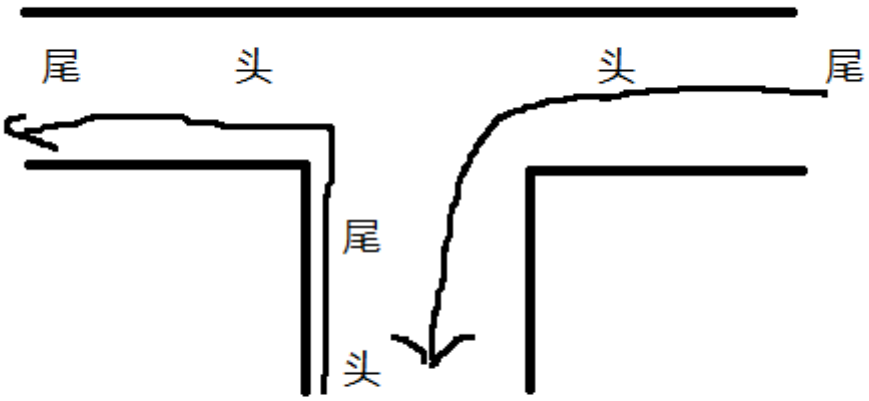
栈（**Stack**）是一种特殊的线性表，它的**插入**和**删除**只允许在线性表的一端进行，称为**栈顶**，不允许操作的一端称作为**栈底**。

先进后出，插入元素称为**Push**，删除元素称为**Pop**。

栈的应用?



> nodejs的中间件



> 火车掉头
>
> 浏览器或者说是app的前进和后退。

抽象数据类型

接口

```
package DS;
public interface Sgstack<T> {
    public boolean isEmpty();   //是否为空
    public void push(T x);      //入栈
    public T peek();            //返回栈顶元素
    public T pop();             //弹出栈顶元素
}
```

顺序栈

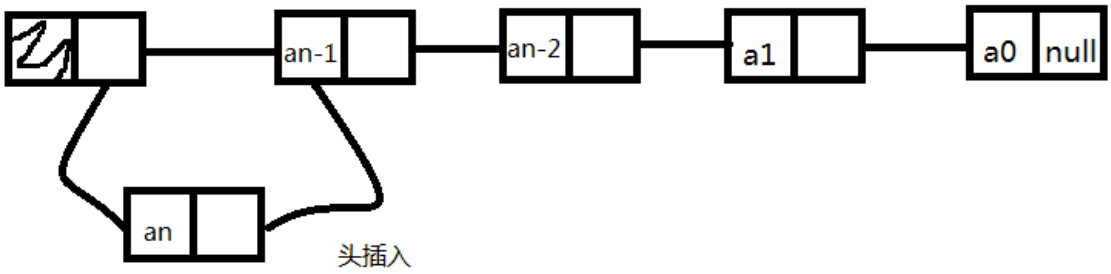
```
package DS;
public class SeqStack<T> implements SqStack<T> {
    int MaxStackSize;
    Object element[];
    int top;
    public SeqStack(int length){
        int top = 0;
        this.MaxStackSize = length;
        this.element = new Object[length];
    }
    public boolean isEmpty(){
        return this.top == 0;
    }
    public void push(T x){
        if(this.top == this.MaxStackSize){
            throw new NullPointerException("数组越界啦");
        }
        this.element[top] = x;
        this.top++;
    }
    public T peek(){
        if(this.top == 0){
            return null;
        }
        return (T)element[top-1];
    }
    public T pop(){
        if(this.top==0){
            throw new NullPointerException("没有元素哦");
        }
        this.top--;
        return (T)element[top];
        //这里看似没有对元素进行删除操作，只是改变了下标，但是事实上我们下一次存储的时候就会覆盖掉上一次要删除的元素。
    }
}

```

> top指的是最大元素上面的元素，就好像压在数据上的一块石头。

链表栈

入栈&&出栈



> 其实就是头插入和头删除。

```java
package DS;
public class SinglyStack<T> implements Sgstack<T> {
    private SinglyList<T> list;
    public SinglyStack(){
        this.list = new SinglyList<T>();
    }
    public boolean isEmpty(){
        return this.list.isEmpty();
    }
    public void push(T x){
        this.list.insert(0,x);
    }
    public T peek(){
        return this.list.get(0);
    }
    public T pop(){
        return this.list.remove(0).data;
    }
}
```

> 我们使用已经实现的单链表类来实现栈，很简单，就是头插入和头删除。

使用栈来实现十进制和二进制的转换

```java
public static String converse(int x){
    SeqStack converseStack = new SeqStack(64);
    while(x!=1){
        if(x%2 == 0){
            converseStack.push(0);
            x = x/2;
        }else{
            converseStack.push(1);
            x = (x-1)/2;
        }
    }
    converseStack.push(1);
    String str = "";
    while(!converseStack.isEmpty()){
        str+=converseStack.pop();
    }
    return str;
}
```

> 想象一下，十进制转换为二进制不就是取余二的过程么，双取0，单取1，最后从最下方在返回来。