数组常用的方法

数组中常见的方法以及如何使用

1.find 查找某个元素

- 返回true则将当前遍历的元素返回
  找出数组中, 名字为jw的项,找到后立即返回不继续查找

```
var arr = [{name:'jw'},{name:'zfpx'},{name:'zry'},{name:'jw'}];
var obj = arr.find(function (item) {
 if(item.name == 'jw'){
     return true;
 }
});
console.log(obj); // jw
```

2.some 查看是否有满足条件的元素

- 返回true则结果为true 查看数组中, 名字是否有jw的项,找到后立即返回不继续查找

```
var arr = [{name:'jw'},{name:'zfpx'},{name:'zry'},{name:'jw'}];
var flag = arr.some(function (item) {
 if(item.name == 'jw'){
     return true;
 }
});
console.log(flag); // true
```

3.map

- 遍历每一项，用返回值替换当前项
  将数组每个人的年龄累加10岁

```
var arr = [{name:'jw',age:18},{name:'zfpx',age:40},{name:'zry',age:32}];
var result = arr.map(function (item) {
  item.age+=10;
  return item
});
console.log(result);
```

4.filter

- 返回true则表示当前项删除，返回false当前项保留
  将数组中年龄小于30岁的删掉

```
var arr = [{name:'jw',age:18},{name:'zfpx',age:40},{name:'zry',age:32}];
var result = arr.map(function (item) {
item.age+=10;
return item
});
console.log(result);
```

5.reduce

- 返回值是经过叠加处理后的操作。
- 函数中第一个参数为prev 第二个参数为next
  求出价格中的总和

```
var arr = [{price:1},{price:2},{price:3}];
var result = arr.reduce(function (prev,next) {
 return prev+next.price;
},0);
console.log(result);
```

6.forEach

- 遍历数组 缺点:不能return循环

```
[1,2,3,4,5].forEach(function (item) {
     if(item==2) return;
});
```

7.for in

- 缺点 1.会遍历所有属性 2.遍历的key是对象数据类型

```
var ary = [1,2,3];
arr.name = 'jw';
for(var key in arr){
 console.log(typeof key);
 console.log(arr[key]);
}
```

8.for of

- 好处:可以break,不会遍历数组以外的属性
- 缺点:不能遍历普通对象

```
var ary = [1,2,3];
ary.name = 'jw';
for(var value of ary){
  if(key == 2){break}
  console.log(ary);
}
```