1、

Given an array of integers, return **indices** of the two numbers such that they add up to a specific target.

You may assume that each input would have ***exactly*** one solution, and you may not use the *same* element twice.

/\*\*

\* @param {number[]} nums

\* @param {number} target

\* @return {number[]}

\*/

var twoSum = function(nums, target) {

var i = 0,anwser=[];

for(var i = 0; i < nums.length; i++){

for(var j = i;j<nums.length;j++){

if(nums[i]+nums[j]==target){

anwser[0]=i;

anwser[1]=j;

}

}

}

return anwser;

};

思路就是从头到尾挨个遍历逐个判断

2、Reverse digits of an integer.

**Example1:** x = 123, return 321  
**Example2:** x = -123, return -321/\*\*

\* @param {number} x

\* @return {number}

\*/

var reverse = function(x) {

var result = 0;

while(x){

result = result \* 10 + x % 10;

x /= 10;

}

if(result<=2147483647||result>=-2147483648){

return result;

}

else{

return 0;

}

};

思路：倒序方面：result从x里取一位放在最右边，再在下一步里把这一位往左移，同时取下一位；之后再判断result是否越界。

3、

* Determine whether an integer is a palindrome. Do this without extra space.

/\*\*

\* @param {number} x

\* @return {boolean}

\*/

var isPalindrome = function(x) {

var result = 0;

while(x){

result = result \* 10 + x % 10;

x /= 10;

}

if(result==x){

return true;

}

else{

return false;

}

};

思路：先求出倒序数字，再判断是否相等

4、Given a sorted array, remove the duplicates in place such that each element appear only *once* and return the new length.

Do not allocate extra space for another array, you must do this in place with constant memory.

For example,  
Given input array *nums* = [1,1,2],

Your function should return length = 2, with the first two elements of *nums* being 1 and 2 respectively. It doesn't matter what you leave beyond the new length.

/\*\*

\* @param {number[]} nums

\* @return {number}

\*/

var removeDuplicates = function(nums) {

for (var i = 0; i < nums.length;i ++){

for(var j = i; j <nums.length; j++){

if(nums[i]==nums[j]){

for(var k = j + 1 ; k < nums.length ; j++,k++){

nums[j]=nums[k];

}

}

}

}

return nums.length;

};

思路：先遍历判断有无重复的，之后用一个for循环将重复的后者之后的所有元素向前一个位置。

5、

Given an array and a value, remove all instances of that value in place and return the new length.

Do not allocate extra space for another array, you must do this in place with constant memory.

The order of elements can be changed. It doesn't matter what you leave beyond the new length.

**Example:**  
Given input array *nums* = [3,2,2,3], *val* = 3

Your function should return length = 2, with the first two elements of *nums* being 2.

/\*\*

\* @param {number[]} nums

\* @param {number} val

\* @return {number}

\*/

var removeElement = function(nums, val) {

for(var i = 0;i < nums.length;i ++){

if(nums[i]==val){

for(var k = i , l = i + 1;l < nums.length; k ++,l ++){

nums[k]=nums[l];

}

nums.pop();

}

}

return nums.length;

};

思路：遇到val就将其后面都都往前移，同时删除最后一位。