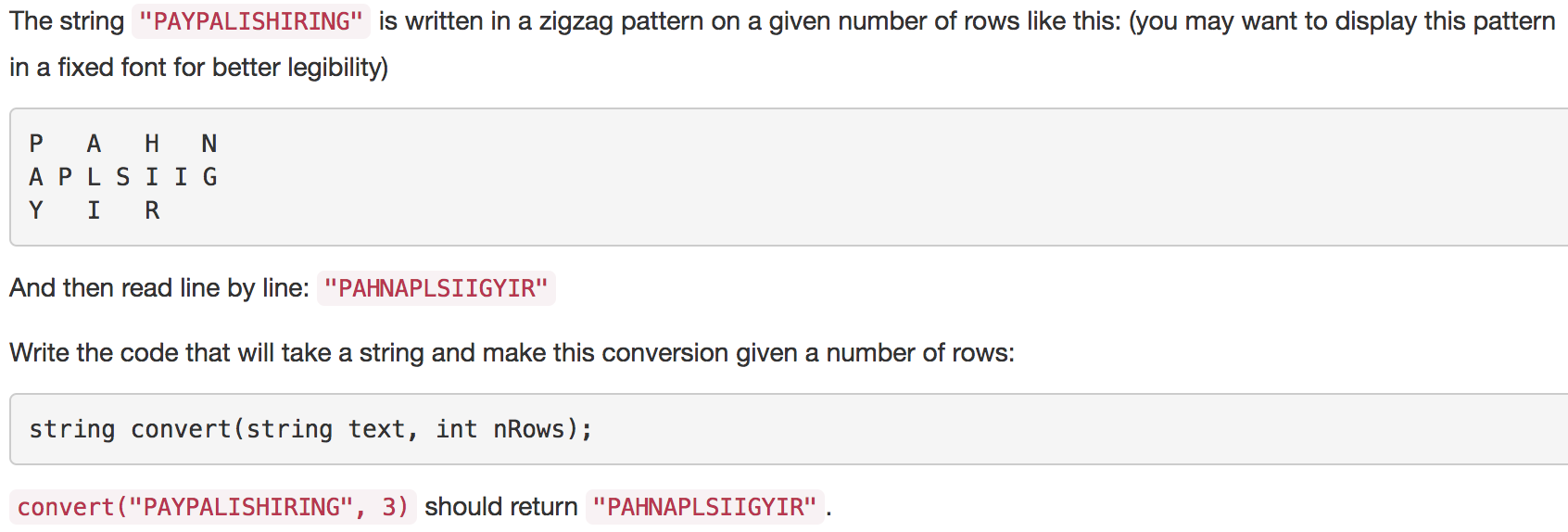
6. ZigZag Conversion[middle]

要求：



思路：定义一个二维数组，然后将Z型结果存储到二维数组中，最后将二维数组内容转换成字符串返回。

/\*\*

\* @param {string} s

\* @param {number} numRows

\* @return {string}

\*/

var convert = function(s, numRows) {

if(numRows>1){

let arr=[];

let c=(numRows-1)\*2;

for(let i=0;i<numRows;i++){

arr[i]=[];

}

for(let i=0;i<s.length;i++){

let row;

if(numRows-1<i%c && i%c<c){

row=c-i%c;

}else{

row=i%c;

}

arr[row][(i-row)/2]=s[i];

}

console.log(arr);

let arr1=[];

for(let i=0;i<arr.length;i++){

for(let j=0;j<arr[i].length;j++){

if(arr[i][j]){

arr1.push(arr[i][j])

}

}

}

res=arr1.join("");

return res;

}else{

return s;

}

};

上面这种方法，将Z型存储进二维数组，以下程序优化了这种方法，定义一个一维数组，每一项存储一个字符串：

/\*\*

\* @param {string} s

\* @param {number} numRows

\* @return {string}

\*/

var convert = function(s, numRows) {

if(numRows>1){

let arr=[];

let c=(numRows-1)\*2;

for(let i=0;i<numRows;i++){

arr[i]="";

}

for(let i=0;i<s.length;i++){

let row;

if(numRows-1<i%c && i%c<c){

row=c-i%c;

}else{

row=i%c;

}

arr[row]+=s[i];

}

return arr.join("");

}else{

return s;

}

};

12. Integer to Roman

要求：将整数转换成罗马数字。

将1-9、10-90、100-900、1000-3000的罗马数字存储在一个二维数组中。

/\*\*

\* @param {number} num

\* @return {string}

\*/

var intToRoman = function(num) {

const arr=[["","I","II","III","IV","V","VI","VII","VIII","IX"],

["","X","XX","XXX","XL","L","LX","LXX","LXXX","XC"],

["","C","CC","CCC","CD","D","DC","DCC","DCCC","CM"],

["","M","MM","MMM"]];

let str="";

str+=arr[3][parseInt(num/1000)];

num%=1000;

str+=arr[2][parseInt(num/100)];

num%=100;

str+=arr[1][parseInt(num/10)];

num%=10;

str+=arr[0][num];

return str;

};