

```

1  /*
2  将一个给定字符串根据给定的行数，以从上往下、从左到右进行 Z 字形排列。
3
4  比如输入字符串为 "LEETCODEISHIRING" 行数为 3 时，排列如下：
5
6  L   C   I   R
7  E T O E S I I G
8  E   D   H   N
9
10 之后，你的输出需要从左往右逐行读取，产生出一个新的字符串，比如："LCIRETOESIIGEDHN"。
11
12 请你实现这个将字符串进行指定行数变换的函数：
13
14  string convert(string s, int numRows);
15
16  示例 1:
17
18  输入: s = "LEETCODEISHIRING", numRows = 3
19  输出: "LCIRETOESIIGEDHN"
20
21  示例 2:
22
23  输入: s = "LEETCODEISHIRING", numRows = 4
24  输出: "LDREOEIIECIHNTSG"
25  解释:
26
27  L       D       R
28  E   O E   I   I
29  E C   I H   N
30  T       S       G
31
32  来源：力扣（LeetCode）
33  链接：https://leetcode-cn.com/problems/zigzag-conversion
34  著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
35  */

```

分析:

- 自己画几个找规律，我找到的规律是每行下标与起始点的关系。

方法一:C++_找规律

```

1  class Solution
2  {
3      public:
4          string convert(string s, int numRows)
5          {
6              if(numRows<=1)
7              {
8                  return s;
9              }
10

```

```

11     string ret_val(s)      ;
12     int     ret_num      = 0 ;
13     int     i            = 0 ;
14     int     count        = 0 ; /*0-偶数次 1-奇数次*/
15     int     temp         = 0 ;
16     // 第0行
17     temp = 0;
18     while(temp < s.size())
19     {
20         ret_val[ret_num++] = s[temp];
21         temp += 2*(numRows-1);
22     }
23     // 第1--> numRows-2 行
24     for(i = 1 ; i < numRows-1;i++)
25     {
26         temp = i ;
27         count = 0 ;
28         while(temp < s.size())
29         {
30             ret_val[ret_num++] = s[temp];
31             if(count==0)
32             {
33                 temp += 2*(numRows-i-1);
34                 count = 1;
35             }
36             else
37             {
38                 temp += 2*i;
39                 count = 0;
40             }
41         }
42     }
43     // 第numRows-1行
44     temp = numRows-1;
45     while(temp < s.size())
46     {
47         ret_val[ret_num++] = s[temp];
48         temp += 2*(numRows-1);
49     }
50
51     return ret_val;
52
53 }
54 };
55
56 /*
57 执行结果:
58 通过
59 显示详情
60 执行用时 :20 ms, 在所有 cpp 提交中击败了51.28% 的用户
61 内存消耗 :10 MB, 在所有 cpp 提交中击败了97.22%的用户
62 */

```

方法二:C++_找规律(代码优化)

```

2  {
3      public:
4          string convert(string s, int numRows)
5          {
6              if(numRows<=1)
7              {
8                  return s;
9              }
10
11             string ret_val(s)                ;
12             int ret_num                      = 0 ;
13             int i                            = 0 ;
14             int count                        = 0 ;/*0-偶数次 1-奇数次,2-顶层或
者底层*/
15             int temp                        = 0 ;
16
17
18             for(i = 0 ; i < numRows;i++)
19             {
20                 temp = i;
21                 count = 0;
22                 if(i == 0 || i == numRows-1)
23                 {
24                     count = 2;
25                 }
26                 while(temp < s.size())
27                 {
28                     ret_val[ret_num++] = s[temp];
29                     if(count==0)
30                     {
31                         temp += 2*(numRows-i-1);
32                         count = 1;
33                     }
34                     else if(count==1)
35                     {
36                         temp += 2*i;
37                         count = 0;
38                     }
39                     else
40                     {
41                         temp += 2*(numRows-1);
42                     }
43                 }
44             }
45             return ret_val;
46
47         }
48 };
49
50 /*
51 执行结果:
52 通过
53 显示详情
54 执行用时 :16 ms, 在所有 cpp 提交中击败了70.03% 的用户
55 内存消耗 :9.8 MB, 在所有 cpp 提交中击败了98.99%的用户
56 */
57

```

