算法讨论班第十二期-王海璐

2015年11月13日星期五

62 Unique Paths

A robot is located at the top-left corner of a *m* x *n* grid (marked 'Start' in the diagram below).

The robot can only move either down or right at any point in time. The robot is trying to reach the bottom-right corner of the grid (marked 'Finish' in the diagram below).

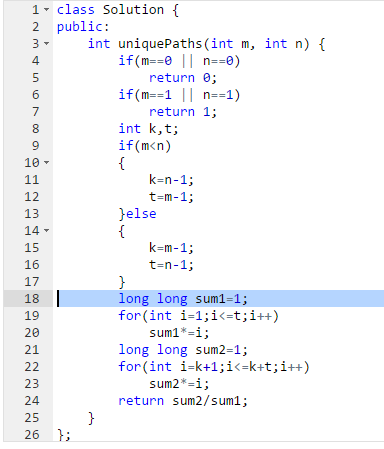
How many possible unique paths are there?



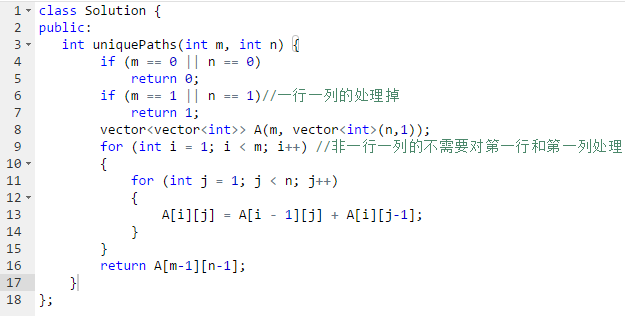
Above is a 3 x 7 grid. How many possible unique paths are there?

**Note:** *m* and *n* will be at most 100.

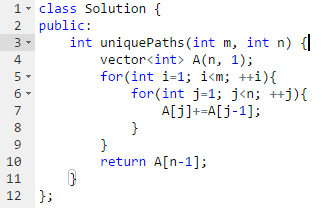
1. 直观思路：
2. 这道题就是说m\*n的格子向右走n-1步，向下走m-1步，总共有多少种不同的路径。依照排列组合的思想，总共需要走m-1+n-1步，中间选择n-1步是向右方向的，其余m-1步是向下方向的（或者中间选择m-1步是向下方向的，其余n-1步是向右方向的），那么总共就是 C(m-1+n-1,  m-1) 或者 C(n-1， m-1+n-1)步。代码计算此排列组合数就可以。
3. 代码注意sum1 sum2要取**long long类型**。不然不够长，第一次错误就是因为取了int 类型。说明Note中提示m、n最大为100是有用的，即你计算阶乘时int会溢出的。



1. 动态规划思路
2. 定义一个二维数组 A[m][n]，其中的元素A[i][j]表示从左上角到位置i行j列总共的路径数。那么我们开始从左上依次计算每元素的值，最后返回 A[m-1][n-1]即是所求的结果；
3. 递推方程是：A[i][j]=A[i-1][j]+A[i][j-1]；A[i-1][j]表示从左上角到它上面一格的总路径数，而可以通过A[i-1][j]向下走一步到A[i][j]这一格， A[i][j-1]表示从左上角到它左边一格的总路径数，而可以通过A[i][j-1]向右走一步到A[i][j]这一格。
4. 初始化A[i][j]均为1；



算法时间复杂度为O（mn），空间复杂度也为O（mn）。可以稍作改进，实现空间复杂度O（n）



63 Unique Paths II

Follow up for "Unique Paths":

Now consider if some obstacles are added to the grids. How many unique paths would there be?

An obstacle and empty space is marked as 1 and 0 respectively in the grid.

For example,

There is one obstacle in the middle of a 3x3 grid as illustrated below.

[

[0,0,0],

[0,1,0],

[0,0,0]

]

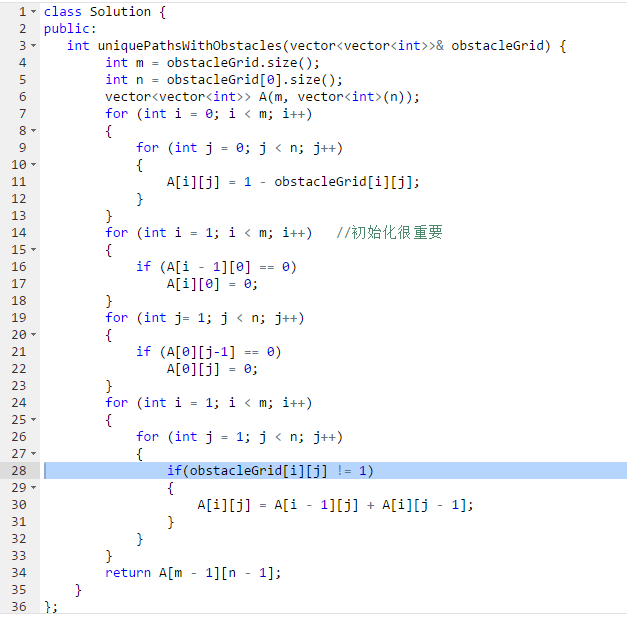
The total number of unique paths is 2.

**Note:** *m* and *n* will be at most 100.

解题思路：

从62题进行改进，注意改进重要的地方

1. 对于所有的格子，如果格子上有障碍物，那么就到不了该点，就直接初始为0,其他格子置为1；
2. 对于第一行和第一列，如果前面到达不了，后面也到达不了，所以在1)的基础上参照前面的路径再次进行初始化首行首列;
3. 循环过程中，只有格子没有障碍物的进行路径计算才是有意义的，所以需要判断当前是否是障碍格子。



65 Valid Number

Validate if a given string is numeric.

Some examples:  
"0" => true  
" 0.1 " => true  
"abc" => false  
"1 a" => false  
"2e10" => true

**Note:** It is intended for the problem statement to be ambiguous. You should gather all requirements up front before implementing one.

1. 自己的思路过程
2. 首先自己想到了利用各种判断，判断‘0’-‘9’之间，‘.’ ‘e’ ‘ ’这些情况的处理，各种判断原则 ‘.’不能重复出现， ‘e’也不能重复出现， ‘ ’不能在出现的位置，然后只要没有不满足这些条件的最后就是合法的。
3. 但是在判断过程中虽例子的情况都是正确结果，提交时候还是遇到了这样那样的奇怪问题。最后整理思路……
4. 重新清晰思路
5. 首先去掉开头和结尾的连续空格，去掉开始的符号字符；
6. 对于每一个字符的情况进行处理：

对于”0”到”9”的数字，合法，并表示之前的数字是合法的；

对于'e'字符，前面必须是合法数字，'e'不能重复，且不能出现在最后；

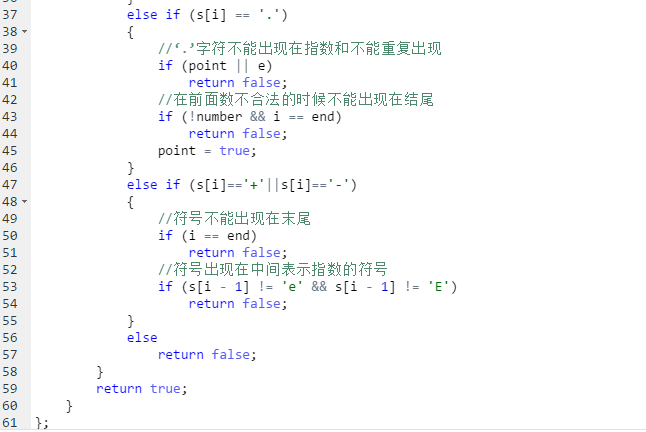
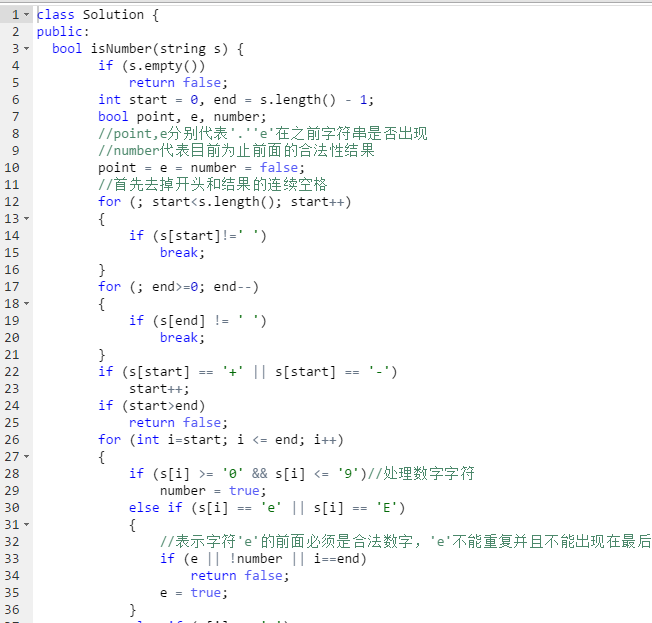
对于‘.’字符不能出现在指数中，不能重复出现，在前面数不合法的时候不能出现在结尾；

对于符号数不能出现在末尾，判断是否是表示指数的符号（符号出现在中间表示指数的符号）；

其余字符出现不合法。

1. 前面的检验通过，就是合法字符。

代码如下：



1. 参考网上

用**有限状态机**，非常简洁，不需要复杂的各种判断！

先枚举一下各种合法的输入情况：

1.空格+ 数字 +空格

2.空格+ 点 + 数字 +空格

3.空格+ 符号 + 数字 +　空格

4.空格 + 符号 + 点 +　数字　＋空格

5.空格 + (1, 2, 3, 4状态） + e +　(1, 2, 3, 4状态) +空格

组后合法的字符可以是：

1.数字

2.空格

有限状态机的状态转移过程：

起始状态为0：

　　当输入空格时，状态仍为0，

　　输入为符号时，状态转为3，3的转换和0是一样的，除了不能再接受符号，故在0的状态的基础上，把接受符号置为-1；

　　当输入为数字时，状态转为1, 状态1的转换在于无法再接受符号，可以接受空格，数字，点，指数；状态1为合法的结束状态；

　　当输入为点时，状态转为2，状态2必须再接受数字，接受其他均为非法；

　　当输入为指数时，非法；

状态1：

　　接受数字时仍转为状态1，

　　接受点时，转为状态4，可以接受空格，数字，指数，状态4为合法的结束状态，

　　接受指数时，转为状态5，可以接受符号，数字，不能再接受点，因为指数必须为整数，而且必须再接受数字；

状态2：

　　接受数字转为状态4；

状态3：

　　和0一样，只是不能接受符号；

状态4：

　　接受空格，合法接受；

　　接受数字，仍为状态4；

　　接受指数，转为状态5，

状态5：

　　接受符号，转为状态6，状态6和状态5一样，只是不能再接受符号，

　　接受数字，转为状态7，状态7只能接受空格或数字；状态7为合法的结束状态；

状态6：

　　只能接受数字，转为状态7；

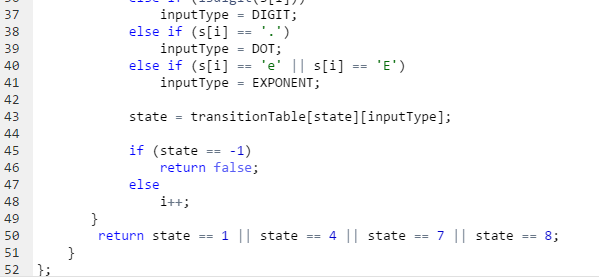
状态7：

　　接受空格，转为状态8，状态7为合法的结束状态；

接受数字，仍为状态7；

状态8：

接受空格，转为状态8，状态8为合法的结束状态；



68 Text Justification

Given an array of words and a length *L*, format the text such that each line has exactly *L* characters and is fully (left and right) justified.

You should pack your words in a greedy approach; that is, pack as many words as you can in each line. Pad extra spaces ' ' when necessary so that each line has exactly*L* characters.

Extra spaces between words should be distributed as evenly as possible. If the number of spaces on a line do not divide evenly between words, the empty slots on the left will be assigned more spaces than the slots on the right.

For the last line of text, it should be left justified and no extra space is inserted between words.

For example,  
**words**: ["This", "is", "an", "example", "of", "text", "justification."]  
**L**: 16.

Return the formatted lines as:

[

"This is an",

"example of text",

"justification. "

]

**Note:** Each word is guaranteed not to exceed *L* in length.

[click to show corner cases.](https://leetcode.com/problems/text-justification/)

**Corner Cases:**

* A line other than the last line might contain only one word. What should you do in this case?  
  In this case, that line should be left-justified.

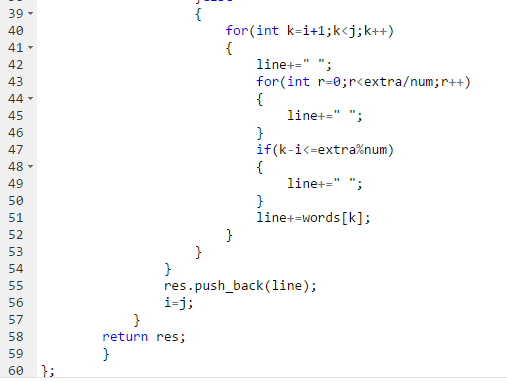
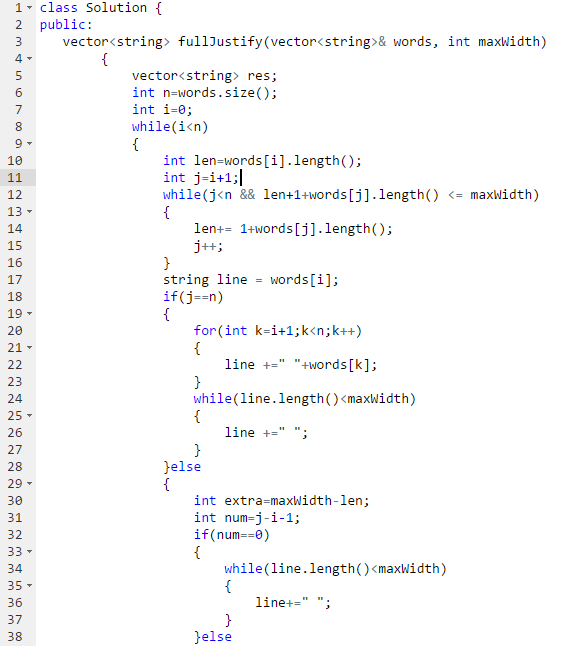
首先题目的意思就相当于我们进行一个英文文本的格式排布，格式规为：左右对齐，单词中间的空格均匀排列，不能够均匀排列的左边的空格数多于右边空格数。

解题思路：

依次构造每一行，分两种情况：

1. 是最后一行：那么每个单词之间一个空格，最后面补空格补该行字符串总长到L为止；
2. 非最后一行：考虑到如果一个单词就只在后面加空格，多个单词就在单词之间加空格，那么就分只有一个单词的和多个单词的情况；
3. 该行只能容纳一个单词，就在单词后面填充空格直到该行字符串总长到L为止；
4. 该行有多个单词（number个），设单词的总字符加上两两单词之间空格的总字符个数达到len个，处理剩余L-len个空格的填充，那么每两个单词之间还需要填充（L-len）/number个空格，对于本行从1到（L-len）%number个原本空格串后面再填充一个空格。
5. 根据前面的情况每次循环构造一行字符串，然后加到结果中。

代码如下：



下期题目:

91，94，140