设计一个算法，找出只含素因子2，3，5 的第 *n* 大的数。

符合条件的数如：1, 2, 3, 4, 5, 6, 8, 9, 10, 12...

 注意事项

我们可以认为1也是一个丑数

您在真实的面试中是否遇到过这个题？

Yes

**样例**

如果n = 9， 返回 10

思路：通过遍历寻找，满足特征就加一，关键在于如何避免重复，通过else if 语句。

#超时

class Solution:

"""

@param {int} n an integer.

@return {int} the nth prime number as description.

"""

def nthUglyNumber(self, n):

# write your code here

k=1

muns=1

iterms=(n\*n)\*n

for i in range(2,iterms):

if panduan(i):

k+=1

muns=i

if k==n:

break

return muns

def panduan(n):

i=n

while i !=1:

if i%2==0:

i=i//2

elif i%3==0:

i=i//3

elif i%5==0:

i=i//5

else:

break

if i==1:

return True

既然一个一个判断超时那么就剩下一个一个造数据了

class Solution {

public:

/\*

\* @param n an integer

\* @return the nth prime number as description.

\*/

int nthUglyNumber(int n) {

// write your code here

int \*ugly = new int[n];

ugly[0] = 1;

int num\_2 = 0;

int num\_3 = 0;

int num\_5 = 0;

for(int i = 1;i<n;i++)

{

ugly[i] = min(min(ugly[num\_2]\*2,ugly[num\_3]\*3),ugly[num\_5]\*5);

if(ugly[i] / ugly[num\_2] == 2)

num\_2 ++;

if(ugly[i] / ugly[num\_3] == 3)

num\_3 ++;

if(ugly[i] / ugly[num\_5] == 5)

num\_5 ++;

}

return ugly[n-1];

}

};

2.给出两个整数a和b, 求他们的和, 但不能使用 + 等数学运算符。

 注意事项

你不需要从输入流读入数据，只需要根据aplusb的两个参数a和b，计算他们的和并返回就行。

您在真实的面试中是否遇到过这个题？

Yes

**说明**

a和b都是 32位 整数么？

* 是的

我可以使用位运算符么？

* 当然可以

**样例**

如果 a=1 并且 b=2，返回3

[**挑战**](http://www.lintcode.com/zh-cn/problem/a-b-problem/#challenge)

这应该也是二进制的加法问题：

class Solution:

"""

@param a: The first integer

@param b: The second integer

@return: The sum of a and b

"""

def aplusb(self, a, b):

# write your code here, try to do it without arithmetic operators

if a==~b+1:

return 0

j=1

while(j):

t=a^b

j=(a&b)<<1

a=t

b=j

return t

3. 设计一个算法，计算出n阶乘中尾部零的个数

您在真实的面试中是否遇到过这个题？

Yes

**样例**

**11! = 39916800**，因此应该返回 2

思路：转换成字符串判断，还是做除法，还是字符串吧

class Solution:

"""

@param: : An integer

@return: An integer, denote the number of trailing zeros in n!

"""

def trailingZeros(self, n):

# write your code here, try to do it without arithmetic operators.

i=1

count=0

while n>=5\*\*i:

count+=n//(5\*\*i)

i+=1

return count

4. 计算数字k在0到n中的出现的次数，k可能是0~9的一个值

您在真实的面试中是否遇到过这个题？

Yes

**样例**

例如n=12，k=1，在**[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]**，我们发现1出现了5次 **(1, 10, 11, 12)**

思路;统计下出现的个数还是以字符串统计即可。

class Solution:

# @param k & n two integer

# @return ans a integer

def digitCounts(self, k, n):

count=0

for i in range(n+1):

for j in range(len(str(i))):

if str(i)[j]==str(k):

count+=1

return count

5. 给出数组 [9,3,2,4,8]，第三大的元素是 4

给出数组 [1,2,3,4,5]，第一大的元素是 5，第二大的元素是 4，第三大的元素是 3，以此类推

线性时间查找算法：

思路：没得说和书上一致即可。分组

def Partition(a, low,high, x):

i=low

high-=1

while (a[i]!= x ) :

i+=1

a[low],a[i]=a[i],a[low]

while (low < high):

while low < high and a[high] >= x:

high-=1

a[low] = a[high]

while low < high and a[low] <x:

low+=1

a[high] = a[low]

a[low] = x

return low

def S\_sort(a,low,high):

n = high - low

if n==1:

return a[low]

remind=n%5

for i in range(0,n//5):

quick\_sort(a, i\*5+low,low+i \* 5 + 4)

a[low+i],a[low+i \* 5 + 2]=a[low+i \* 5 + 2], a[low+i]

h=n//5

if remind !=0:

quick\_sort(a, high-remind,high-1)

a[low+h],a[high-remind+(remind+1)//2-1]=a[high-remind+(remind+1)//2-1], a[low+h]

x = S\_sort(a, low,low+h+1)

else:

x = S\_sort(a, low,low+h)

return x

def Select(a, low,high, k):

x=S\_sort(a, low, high)

j= Partition(a, low, high, x)

q = j - low + 1

if (q == k) :

return x

elif (q>k):

return Select(a, low, j , k)

else :

return Select(a, j+1, high, k - q)

def quick\_sort(lists, left, right):

# 快速排序

if left >= right:

return lists

key = lists[left]

least = left

max1 = right

while left < right:

while left < right and lists[right] >= key:

right -= 1

lists[left] = lists[right]

while left < right and lists[left] <= key:

left += 1

lists[right] = lists[left]

lists[right] = key

quick\_sort(lists,least, left - 1)

quick\_sort(lists, left + 1, max1)

return lists

a = [ 8, 4,0, -89, -12, 1, 36, 789, 21, 54,2,6,5,9,39,24,0]

t= Select(a, 0,len(a),10)

print(t)

6. 合并两个排序的整数数组A和B变成一个新的数组。

您在真实的面试中是否遇到过这个题？

Yes

**样例**

给出A=**[1,2,3,4]**，B=**[2,4,5,6]**，返回**[1,2,2,3,4,4,5,6]**

**思路：扑克牌排序：**

class Solution:

#@param A and B: sorted integer array A and B.

#@return: A new sorted integer array

def mergeSortedArray(self, A, B):

# write your code here

m=len(A)

n=len(B)

A.append(8888)

B.append(8888)

c=[]

for i in range(m+n):

c.append(0)

i=0;j=0

for k in range(m+n):

if A[i] < B[j]:

c[k]=A[i]

i+=1

else:

c[k]=B[j]

j+=1

return c

7. 设计一个算法，并编写代码来序列化和反序列化二叉树。将树写入一个文件被称为“序列化”，读取文件后重建同样的二叉树被称为“反序列化”。

如何反序列化或序列化二叉树是没有限制的，你只需要确保可以将二叉树序列化为一个字符串，并且可以将字符串反序列化为原来的树结构。

 注意事项

There is no limit of how you deserialize or serialize a binary tree, LintCode will take your output of serialize as the input of deserialize, it won't check the result of serialize.

您在真实的面试中是否遇到过这个题？

Yes

**样例**

给出一个测试数据样例， 二叉树{3,9,20,#,#,15,7}，表示如下的树结构：

3

/ \

9 20

/ \

15 7

我们的数据是进行BFS遍历得到的。当你测试结果wrong answer时，你可以作为输入调试你的代码。

你可以采用其他的方法进行序列化和反序列化。

思路：根据观察序列化和反序列化就是二叉树的创建与遍历。

序列化就是层次遍历。还是老办法做个level表示同一层，关键在于空节点的表达。出口有哪些，还有必须要知道树深是多少。或者判断一下什么情况跳出循环。最好还是做下树深。下面就是怎么把list变成树了，list变成树一般是递归建立。

"""

Definition of TreeNode:

class TreeNode:

def \_\_init\_\_(self, val):

self.val = val

self.left, self.right = None, None

"""

class Solution:

'''

@param root: An object of TreeNode, denote the root of the binary tree.

This method will be invoked first, you should design your own algorithm

to serialize a binary tree which denote by a root node to a string which

can be easily deserialized by your own "deserialize" method later.

'''

def serialize(self, root):

# write your code here

if root==None:

return None

temp=[]

b=[]

q=[]

for i in range(100):

if not self.temp2(root,i,temp):

break

b.append(temp)

temp=[]

for i in b:

for j in i:

q.append(j)

while q[-1]=='#':

q.pop()

return q

def temp2(self,root,level,temp):

if level <0:

return 0

if level>0 and not root:

return 0

if level==0:

if root==None:

temp.append('#')

return 0

else:

temp.append(root.val)

return 1

return self.temp2(root.left,level-1,temp) + self.temp2(root.right,\

level-1,temp)

'''

@param data: A string serialized by your serialize method.

This method will be invoked second, the argument data is what exactly

you serialized at method "serialize", that means the data is not given by

system, it's given by your own serialize method. So the format of data is

designed by yourself, and deserialize it here as you serialize it in

"serialize" method.

'''

def deserialize(self, data):

# write your code here

if data==None:

return None

else:

root=TreeNode(data[0])

self.gettree(root,data,0)

return root

def gettree(self,root,data,i):

if data[i]==None:

return

if data[i]!=None:

root.val=data[i]

if 2\*i+1<len(data):

root.left=TreeNode(0)

self.gettree(root.left,data,2\*i+1)

if 2\*i+2<len(data):

root.right=TreeNode(0)

self.gettree(root.right,data,2\*i+2)

8. 给定一个字符串和一个偏移量，根据偏移量旋转字符串(从左向右旋转)

您在真实的面试中是否遇到过这个题？

Yes

**样例**

对于字符串 "abcdefg".

offset=0 => "abcdefg"

offset=1 => "gabcdef"

offset=2 => "fgabcde"

offset=3 => "efgabcd"

思路：加入偏移量大于字符串长度应该怎么处理。所以应该是最后用的应该是余数。

class Solution:

# @param s: a list of char

# @param offset: an integer

# @return: nothing

def rotateString(self, s, offset):

# write you code here

length=len(s)

if length==0:

length=1

offset=offset%length

t=s[0:length-offset]

del(s[0:length-offset])

s+=t

9. 给你一个整数*n*. 从 *1* 到 *n* 按照下面的规则打印每个数：

* 如果这个数被3整除，打印fizz.
* 如果这个数被5整除，打印buzz.
* 如果这个数能同时被3和5整除，打印fizz buzz.

您在真实的面试中是否遇到过这个题？

Yes

**样例**

比如 *n* = 15, 返回一个字符串数组：

[

"1", "2", "fizz",

"4", "buzz", "fizz",

"7", "8", "fizz",

"buzz", "11", "fizz",

"13", "14", "fizz buzz"

]

思路：同时放在最开始判断，其余的放在下面判断这样才不会有错误。

class Solution:

"""

@param n: An integer as description

@return: A list of strings.

For example, if n = 7, your code should return

["1", "2", "fizz", "4", "buzz", "fizz", "7"]

"""

def fizzBuzz(self, n):

results = []

for i in range(1, n+1):

if i % 15 == 0:

results.append("fizz buzz")

elif i % 5 == 0:

results.append("buzz")

elif i % 3 == 0:

results.append("fizz")

else:

results.append(str(i))

return results

10．给定两个值 k1 和 k2（k1 < k2）和一个二叉查找树的根节点。找到树中所有值在 k1 到 k2 范围内的节点。即打印所有x (k1 <= x <= k2) 其中 x 是二叉查找树的中的节点值。返回所有升序的节点值。

您在真实的面试中是否遇到过这个题？

Yes

**样例**

如果有 k1 = 10 和 k2 = 22, 你的程序应该返回 [12, 20, 22].

20

/ \

8 22

/ \

4 12

思路一：就是把树转换成list然后查找一下就知道，思路二直接在树上直接查找吧。出口就是左边的没有k1大的，右边没有比k2小的。既然添加是还要注意是中序才能保证大小，不然还要排序。

"""

Definition of TreeNode:

class TreeNode:

def \_\_init\_\_(self, val):

self.val = val

self.left, self.right = None, None

"""

class Solution:

"""

@param root: The root of the binary search tree.

@param k1 and k2: range k1 to k2.

@return: Return all keys that k1<=key<=k2 in ascending order.

"""

def searchRange(self, root, k1, k2):

# write your code here

b=[]

search(root,k1,k2,b)

return b

def search(root,k1,k2,b):

if k1>k2 or root==None:

return b

if root.val>k2:

search(root.left,k1,k2,b)

if root.val<k1:

search(root.right,k1,k2,b)

if root.val>=k1 and root.val<=k2:

search(root.left,k1,k2,b)

b.append(root.val)

search(root.right,k1,k2,b)