191

class Solution(object):

def hammingWeight(self, n):

"""

:type n: int

:rtype: int

"""

sum = 0

while n!=0:

sum += n&1

n = n >> 1

return sum

201

class Solution(object):

def rangeBitwiseAnd(self, m, n):

"""

:type m: int

:type n: int

:rtype: int

"""

i = 0

while m != n:

m = m >> 1

n = n >> 1

i += 1

return m << i

338

class Solution(object):

def num\_of\_1(self, i):

sum = 0

while i != 0:

sum += i&1

i >>= 1

return sum

def countBits(self, num):

"""

:type num: int

:rtype: List[int]

"""

res = [0]

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

s = self.num\_of\_1(i)

res.append(s)

return res

162

class Solution(object):

def findPeakElement(self, nums):

"""

:type nums: List[int]

:rtype: int

"""

l = 0

r = len(nums) - 1

while l < r:

mid = ( l + r ) // 2

if nums[mid] < nums[mid + 1]:

l = mid + 1

else:

r = mid

return l