**K=2**

**2.3.1**

100100010

10001111

11000001001

**2.3.5**

01111100

**2.3.9**

按科学记数法存放：110.1=1.101\*2^010

最高位为符号位110.1（2）为正数，故s位为0，之后三位为指数部分，本题中为010，再向后，整数部分为1，不存储，最后四位存储101，补零，1010

故为 00101010

**K=0**

**2.5.1**

1KB=8192b

**2.5.3**

536.87GB

**2.5.5**

6c99

8001

5e08

**2.5.7**

o

?

**2.1**

a = input('请输入一个二进制数:')

def dec(k,b=0):

if len(k) == 1:

if k == '1':

b = '1'

return b

else:

if k[0] == '1':

b += 2\*\*(len(k)-1)

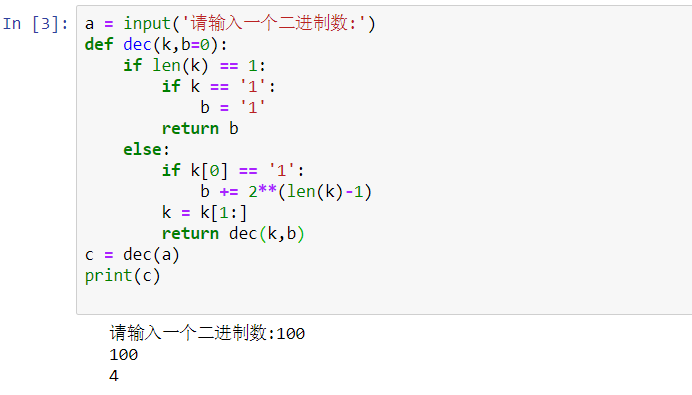
k = k[1:]

return dec(k,b)

print(a)

c = dec(a)

print(c)



**2.10（b）**

def conver\_1(x): # 十转二进制

if x<2:

return([x])

r=x%2

return(conver\_1(x//2)+[r])

x = int(input('输入一个无符号整数：'))

y = int(input('输入最大位数：'))

x2 = abs(x)

x1 = conver\_1(x2)

k = []

if not (- 2 \*\* (y - 1) <= x and x < 2 \*\* (y - 1)):

print('error')

else:

while len(x1) < y: #补零

x1 = [0]+x1

if x>=0:

for i in x1:

print(i,end='')

else:

for i1 in range(len(x1)):

k.append(x1[i1])

else:

for i2 in range(len(k)): #取反码

if k[i2] == 0:

k[i2]= 1

else:

k[i2]= 0

flag = True

for i3 in range(len(k)-1,-1,-1): #加一运算

if flag == True and k[i3] == 0:

k[i3] = 1

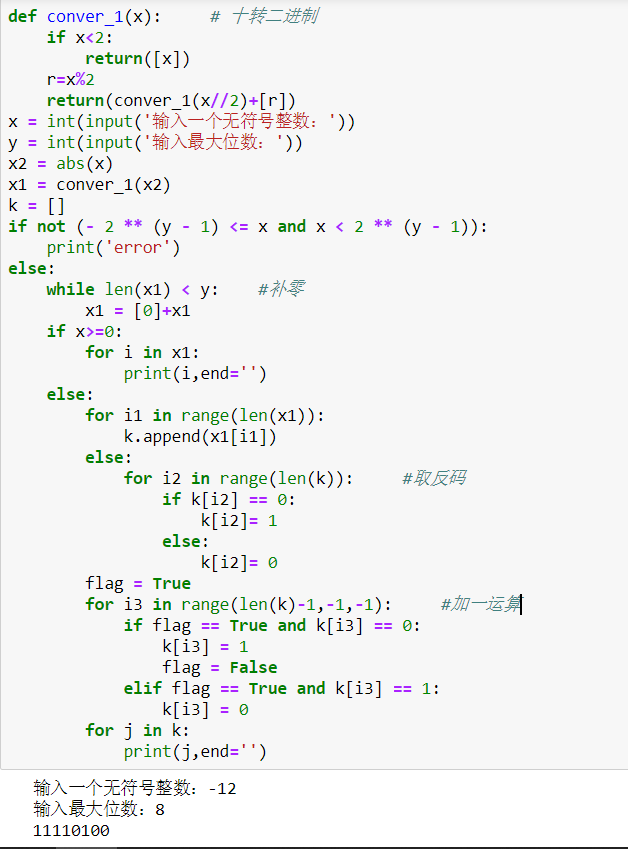
flag = False

elif flag == True and k[i3] == 1:

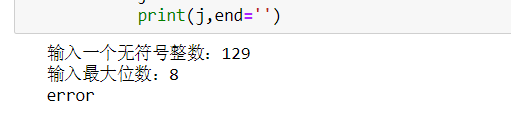
k[i3] = 0

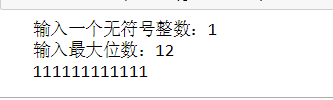
for j in k:

print(j,end='')











**2.7(c)** def g(x):

d = 0

weight = 2\*\*(len(x)-2)

for i in range(1,len(x)):

if x[i] == '1':

d = d + weight

weight = weight // 2

return (d)

def f1(x):

flag = True

x1 = x[:]

for i in range(len(x)-1,-1,-1):

if flag == True and x1[i] == 1:

x1[i] = 0

flag = False

elif flag == True and x1[i] == 0:

x1[i] = 1

for i2 in range(len(x)): #取反码

if x1[i2] == 0:

x1[i2]= 1

elif x1[i2] == 1:

x1[i2]= 0

return x1

def f(x):

d = ''

k =[]

if not (-128) <= x and x < 128:

print('error')

else:

x1 = conver\_1(abs(x))

while len(x1) < 8: #补零

x1 = [0]+x1

if x>=0:

for i in x1:

d = d + str(i)

return d

else:

for i1 in range(len(x1)):

k.append(x1[i1])

else:

for i2 in range(len(k)): #取反码

if k[i2] == 0:

k[i2]= 1

else:

k[i2]= 0

flag = True

for i3 in range(len(k)-1,-1,-1): #加一运算

if flag == True and k[i3] == 0:

k[i3] = 1

flag = False

elif flag == True and k[i3] == 1:

k[i3] = 0

for j in k:

d = d + str(j)

return d

def conver\_1(x): #十进制转二进制

if x<2:

return([x])

r=x%2

return(conver\_1(x//2)+[r])

def add(a,b): #使用字符串补位等长加法

if len(a) < len(b):

b,a = a,b

a = a[::-1] #倒序输出字符串

b = b[::-1]

while len(a) != len(b): #补位

b = b + "0"

extra = 0 #用于进位运算

new = ""

for i, j in enumerate(a):

b\_sum = int(b[i])

new = new + str((int(j) + b\_sum + extra) % 2) #加法

if int(j) + b\_sum + extra > 1: #是否进位

extra = 1

else:

extra = 0

if extra == 1: #最高位是否进位

new = new + '1'

return new[::-1]

a = int(input('请输入一个介于-128到127的整数：'))

b = int(input('请输入一个介于-128到127的整数：'))

if a < -128 or a > 127 or b < -128 or b > 127:

print('数据不合法请输入一个介于-128到127的整数')

else:

a1 = f(a)

b1 = f(b)

c = add(a1,b1)

c1 = c[::-1]

if a\*b <= 0:

if abs(a) == abs(b):

print('0')

elif c[0] == '1':

m = f1(c)

print(int(g(m))-128)

elif c[0] == '0':

print(g(c))

elif a > 0 and b > 0 and c[0] == '1':

print("溢出")

elif a < 0 and b < 0 and c1[7] == '0':

print("溢出")

elif a < 0 and b < 0 and c1[7] != '0':

a2 = f(-a)

b2 = f(-b)

c2 = add(a2,b2)

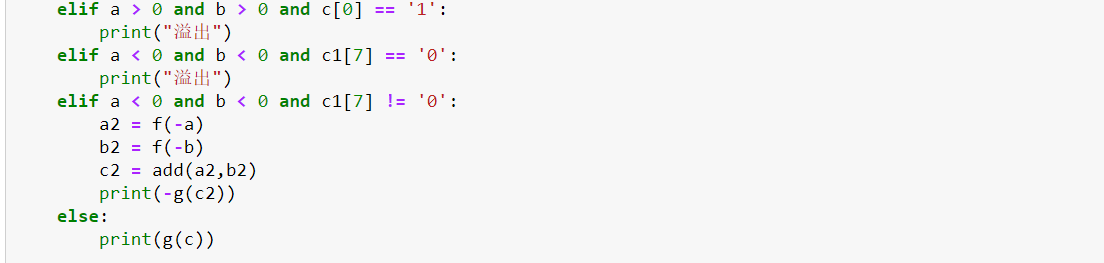
print(-g(c2))

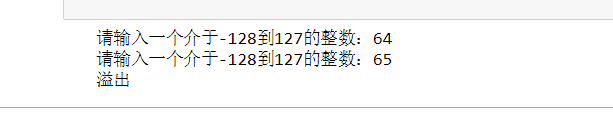
else:

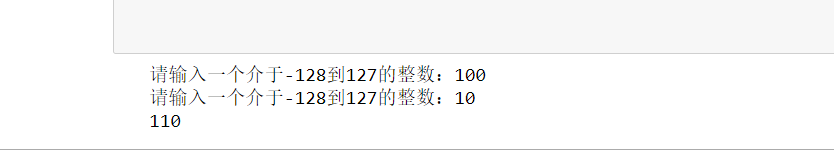
print(g(c))

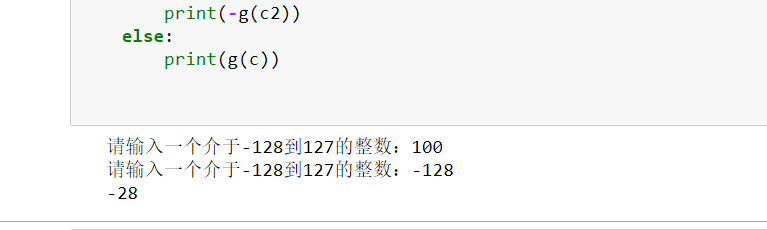


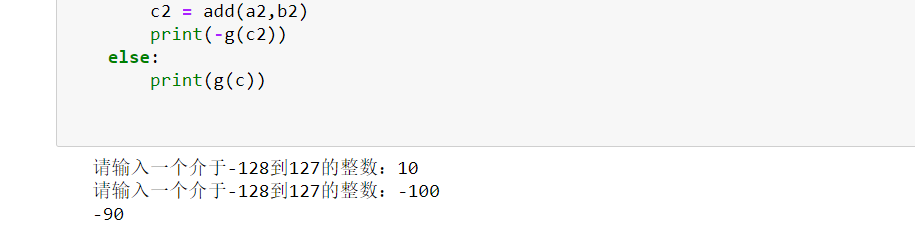


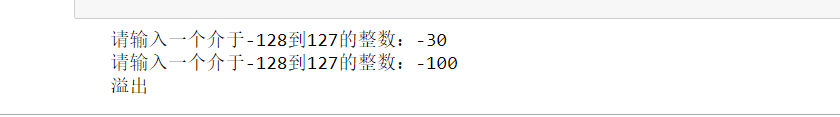


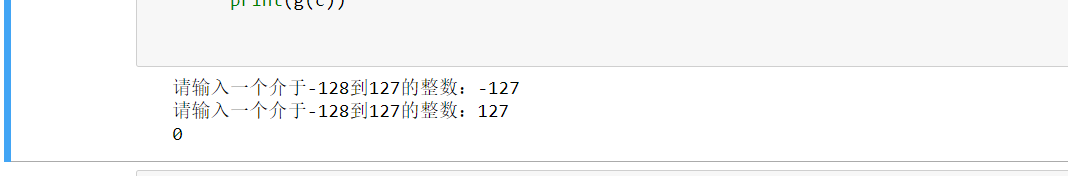




****

****

****

****

**4A**

def g(x): #二进制转十进制

d = 0

weight = 2\*\*(len(x)-1)

for i in range(0,len(x)):

if x[i] == '1':

d = d + weight

weight = weight // 2

return d

def add(a,b): #使用字符串补位等长加法

if len(a) < len(b):

b,a = a,b

a = a[::-1] #倒序输出字符串

b = b[::-1]

while len(a) != len(b): #补位

b = b + "0"

extra = 0 #用于进位运算

new = ""

for i, j in enumerate(a):

b\_sum = int(b[i])

new = new + str((int(j) + b\_sum + extra) % 2) #加法

if int(j) + b\_sum + extra > 1: #是否进位

extra = 1

else:

extra = 0

if extra == 1: #最高位是否进位

new = new + '1'

return new[::-1]

def conver\_1(x): #十进制转二进制

if x<2:

return([x])

r=x%2

return(conver\_1(x//2)+[r])

def mul(a,b):

L = ''

S1 = ''

S2 = ''

a1 = conver\_1(a)

b1 = conver\_1(b)

for i1 in a1:

S1=S1+str(i1)

for j1 in b1:

S2=S2+str(j1)

if len(a1)<len(b1):

S1,S2 = S2,S1

for i in range(len(S2)):

if S2[len(S2)-i-1] == '1':

l = S1 + '0'\*i

L = add(L, l)

return L

a = int(input('请输入一个正整数：'))

b = int(input('请输入一个正整数：'))

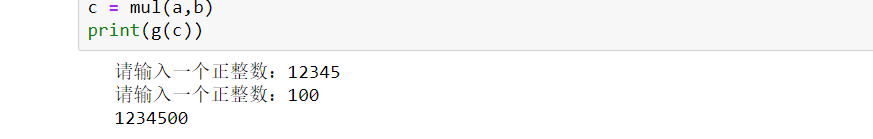
c = mul(a,b)

print(c)

print(g(c))







**4B （本题询问了魏明达一些结构）**

def g(x): #二进制转十进制

d = 0

weight = 2\*\*(len(x)-1)

for i in range(0,len(x)):

if x[i] == '1':

d = d + weight

weight = weight // 2

return d

def conver\_1(x): #十进制转二进制

if x<2:

return([x])

r=x%2

return(conver\_1(x//2)+[r])

def fu(L):

for i in range(len(L)):

if L[i] == 1:

L[i] = 0

else:

L[i] = 1

k = True

for i in range(len(L)-1, -1, -1):

if k == True and L[i] == 0:

L[i] = 1

k = False

elif k == True and L[i] == 1:

L[i] = 0

return L

def add(a,b): #使用字符串补位等长加法

if len(a) < len(b):

b,a = a,b

a = a[::-1] #倒序输出字符串

b = b[::-1]

while len(a) != len(b): #补位

b = b + "0"

extra = 0 #用于进位运算

new = ""

for i, j in enumerate(a):

b\_sum = int(b[i])

new = new + str((int(j) + b\_sum + extra) % 2) #加法

if int(j) + b\_sum + extra > 1: #是否进位

extra = 1

else:

extra = 0

if extra == 1: #最高位是否进位

new = new + '1'

return new[::-1]

def div(a,b): #除法

a1 = conver\_1(a)

b1 = conver\_1(b)

L1 = []

L2 = []

for i in a1:

L1.append(i)

for j in b1:

L2.append(j)

k = 0

while 1:

s1 = ''

s2 = ''

x1 = [0]\*(len(L1)-len(L2)+1)+L2

#补零

f1 = fu(x1)

f2 = [0] + L1

for i1 in f1:

s1 = s1 + str(i1)

for j1 in f2:

s2 = s2 + str(j1)

f = add(s1,s2)

f3 = list(f)

g = 1

for j in range(1, len(f)): #

if j == 1:

g1 = 0

break

if len(L1) < len(L2) or f[1] == '1' or g1:

break

a2 = len(L1)

L1 = f3

if len(L1) > a2:

L1.remove(L1[0])

while not int(L1[0]) == 1 and len(L1)-1 == 1:

L1.remove(L1[0])

k += 1

return k,L1

c = ''

d = ''

a = int(input('请输入被除数：'))

b = int(input('请输入除数：'))

k,L1 = div(a,b)

for j2 in L1:

d = d + str(j2)

print('商=',k)

print('余数=',g(d))

