**Kulvir Singh**

**19BCE2074**

**Experiment 4 (01.03.2021)**

*Question 1*

***Problem Statement :***

Write a python program to perform the following encoding and decoding for the even numbers from 1 – 20.

1. Elias Gamma
2. Elias Delta
3. Golomb (b = 10)

***Procedure :***

Elias Gamma Encoding

Write *x* in binary. Subtract 1 from the number of bits written in step 1 and prepend that many zeros. Use basic python libraries and functions to perform the same.

Elias Gamma Decoding

Read and count zeroes from the stream until we reach the first one. Call this count of zeroes *K*. Consider the one that was reached to be the first digit of the integer, with a value of 2*K*, read the remaining *K* bits of the integer. Use basic python libraries and functions to perform the same.

Elias Delta Encoding

Let N = ⌊log2 X⌋; be the highest power of 2 in X, so 2N ≤ X < 2N+1. Let L = ⌊log2 N+1⌋ be the highest power of 2 in N+1, so 2L ≤ N+1 < 2L+1. Write L zeros, followed by the L+1-bit binary representation of N+1, followed by all but the leading bit (i.e. the last N bits) of X.

Elias Delta Decoding

Read and count zeros from the stream until you reach the first one. Call this count of zeros L. Considering the one that was reached to be the first digit of an integer, with a value of 2L, read the remaining L digits of the integer. Call this integer N+1, and subtract one to get N. Put a one in the first place of our final output, representing the value 2N. Read and append the following N digits.

Golomb Encoding

To Golomb-code a number, find the quotient and remainder of division by the divisor. Write the quotient in unary notation, then the remainder in truncated binary notation. In practice, you need a stop bit after the quotient: if the quotient is written as a sequence of zeroes, the stop bit is a one (or vice versa - and people do seem to prefer to write their unary numbers with ones, which is Wrong). The length of the remainder can be determined from the divisor.

Golomb Decoding

k ←⌈log2(m)⌉. t ← 2k − m. Let s ← the number of consecutive ones in the input (we stop when we read a 0). Let x ← the next k − 1 bits in the input. If x < t: s ← s × m + x. Else: x ← x × 2+ next input bit. s ← s × m + x − t.

***Code :***

Elias Gamma

from math import log

log2 = lambda x: log(x, 2)

def Unary(x):

    return (x-1)\*'0'+'1'

def Binary(x, l = 1):

    s = '{0:0%db}' % l

    return s.format(x)

def binaryToDecimal(binary):

    binary1 = binary

    decimal, i, n = 0, 0, 0

    while(binary != 0):

        dec = binary % 10

        decimal = decimal + dec \* pow(2, i)

        binary = binary//10

        i += 1

    return decimal

def Elias\_Gamma(x):

    if(x == 0):

        return '0'

    n = 1 + int(log2(x))

    b = x - 2\*\*(int(log2(x)))

    l = int(log2(x))

    return Unary(n) + Binary(b, l)

ec=[]

even\_num = [2,4,6,8,10,12,14,16,18,20]

for i in even\_num:

  ec.append(Elias\_Gamma(i))

  print('Elias Gamma of '+str(i)+' : '+(Elias\_Gamma(i)))

for i in ec:

  binstr = (i[i.index('1'):])

  binstr = int(binstr)

  print('Original Value of '+str(i)+' : '+str(binaryToDecimal(binstr)))

Elias Delta

import math

def gamma(t):

    x=[];

    y=[];

    while(t>0):

        x.append(t%2);

        t=int(t/2);

    for i in range(len(x)-1):

        y.append(0);

    for i in range(len(x)):

        y.append(x.pop());

    return y;

even\_num = [2,4,6,8,10,12,14,16,18,20]

ed = []

for x in even\_num:

 temp = str(x)

 t=math.floor(1+math.log(x,2));

 p=gamma(t);

 y=[];

 while(x>0):

     y.append(x%2);

     x=int(x/2);

 y.pop();

 for i in range(len(y)):

     p.append(y.pop());

 a=''.join(map(str,p));

 ed.append(a)

 print('Elias Delta Code of '+temp+' : '+a)

 def decode(x):

    num=0;

    for i in range(len(x)):

        num+=(int(x[len(x)-1-i])\*(math.pow(2,i)));

    return num;

for x in ed:

 if(x=='1'):

     print('1');

     exit;

 else:

     x=list(x);

     t=0;

     v=[];

     b=0;

     w=[];

     c=0;

     for i in x:

         if(b!=1):

             if(i=='0'):

                 t+=1;

             else:

                 v.append(i);

                 b=1;

         elif(c!=1):

             if(t==0):

                 c=1;

                 w.append('1');

                 w.append(i);

             else:

                 v.append(i);

                 t-=1;

         else:

             num=decode(v);

             if(num==0):

                 break;

             else:

                 w.append(i);

                 num-=1;

     ans=decode(w);

     print('Original Number: ',int(ans));

Golomb Code:

import math

def listToString(l) :

  s=""

  for i in l:

    s = s+str(i)

  return s

def encode(t):

    x=[];

    if(t==0):

        return [0];

    while(t>0):

        x.append(t%2);

        t=int(t/2);

    return x

def unary(t):

    y=[];

    for i in range(t-1):

        y.append(0);

    y.append(1)

    return y;

GolombCode = []

even\_num=[2,4,6,8,10,12,14,16,18,20]

b=10

for x in even\_num:

 q=int(x/b)

 y=unary(q+1)

 r=x-(q\*b)

 i=math.floor(math.log(b,2));

 d=math.pow(2,i+1)-b;

 if(r>=d):

     r+=int(d);

 r2=encode(r);

 if(len(r2)<=i and r>=d):

     r2.append(0);

 if(len(r2)<i and r<d):

     r2.append(0);

 r2=r2[::-1];

 y=y+r2;

 GolombCode.append(listToString(y));

 print('Code of '+str(x)+': '+listToString(y))

def decode(x):

    num=0;

    for i in range(len(x)):

        num+=(int(x[len(x)-1-i])\*(math.pow(2,i)));

    return num;

for i in range(10):

 x=str(input('Enter code: '))

 x=list(x)

 i=math.floor(math.log(b,2))

 d=math.pow(2,i+1)-b

 p2=0;

 l=1;

 while(p2<len(x)):

     t=0;

     flag=0;

     r=[];

     k=i;

     q=0;

     for p in range(p2,len(x)):

         if(x[p]=='0' and flag==0):

             t+=1;

             continue;

         if(x[p]=='1' and flag==0):

             q=t;

             flag=1;

             continue;

         r.append(x[p]);

         k-=1;

         if(k==0):

             rnum=decode(r);

             if(rnum<d):

                 p2=p+1;

                 break;

         if(k==-1):

             rnum=decode(r);

             rnum=rnum-d;

             p2=p+1;

             break;

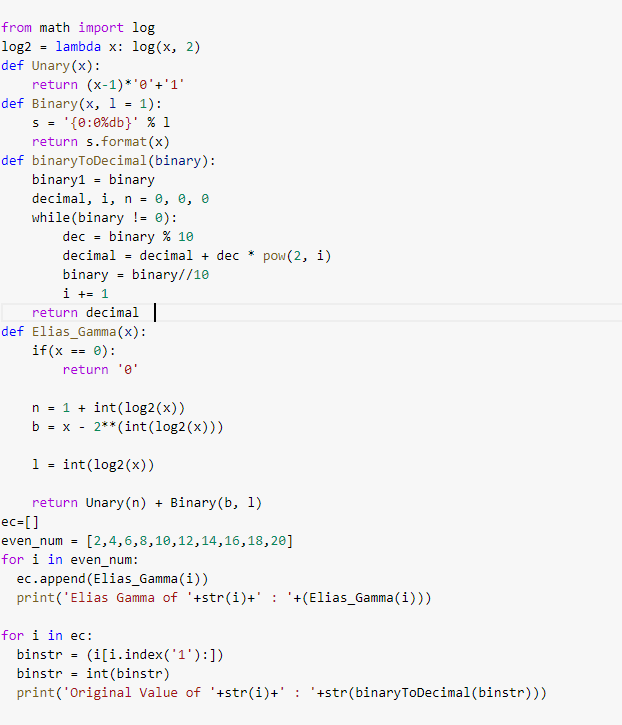
     ans=q\*b+rnum;

     print(ans);

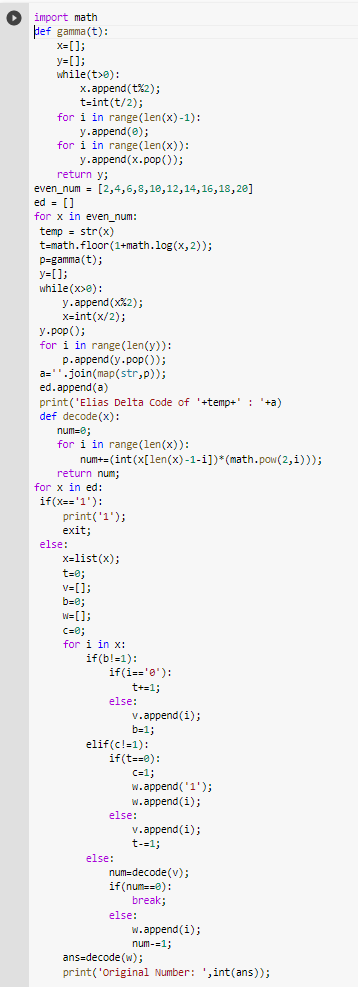
     l=0;

***Code Screenshot:***

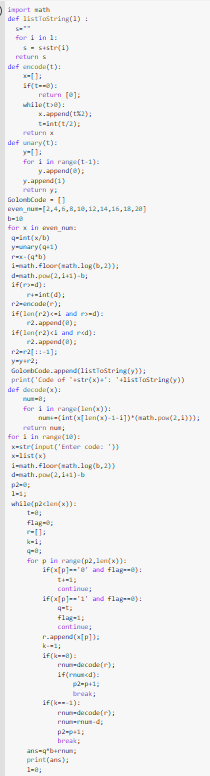
Elias Gamma



Elias Delta

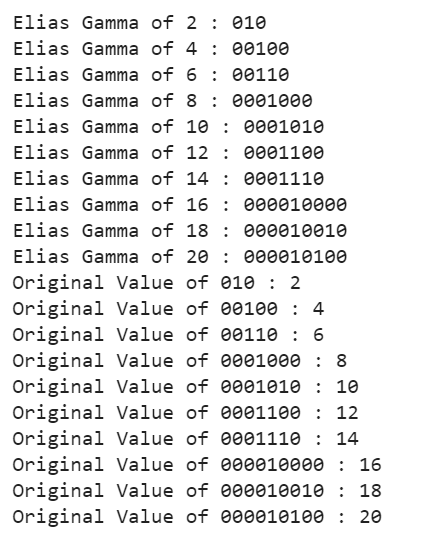


Golomb Code:

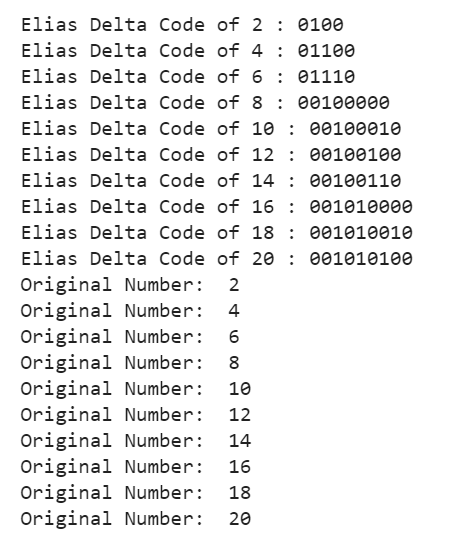


***Output Screenshots :***

***Elias Gamma***



***Elias Delta***



***Golomb Code:***

