Spring compression

Spring Algorithm Introduce:

I want introduce new compression I called Spring compression I didn’t find such compression I get up this algorithm compression and name it Spring. This spring compression book written by Jurijus Pacalovas.

Spring written in Python.

Here is my algorithm Spring:

```
Number_of_the_file=((((Number_of_the_file*2**y)+Add)//3)*M)//Divided

V

Divided_corrdiates=int(University_file[0:(X2*8)],2) Times_12=int(University_file[(X2*8):(X2*8)+8],2)
Times_10=int(University_file[(X2*8)+8:(X2*8)+16],2)
Times_11=int(University_file[(X2*8)+16:(X2*8)+24],2)
N_5=int(University_file[(X2*4)+24:(X2*4)+32],2)
Times_7=int(University_file[(X2*8)+32:(X2*8)+40],2)

X+=1

counts+=1

Times repeat this

check equal numbers to Number_of_the_file

size of long

and counts

spin it up

k1+=1

k2+=1

X2=X1

C11="0"+str(((4*X2)+20))+"b"

if University>(2**((4*X1)+20)-1):

University=0

k1=-1

k2=0 counts=-1

X1+=1

if X1>XN:

University=0

X1=1
```

```
X2=1
```

```
University_file=format(University,C11)
```

V

save it:

```
lenf=len(File_information5_17) Time_Real3=bin(lenf2)[2:] T1=len(Time_Real3)
Time_Real4=format(T1,'06b') long_file=Time_Real4+Time_Real3 C1=bin(X1)[2:] C5=len(C1) C2=C5//8
C4=C5%8 if C4!=0: C3=(C2+1)*8 else: C3=C2*8 C="0"+str(C3)+"b" Time_Real3=format(X1,C)
T1=len(Time_Real3) Time_Real1=bin(T1)[2:] T2=len(Time_Real1) Time_Real4=format(T2,'06b')
XN=Time_Real4+Time_Real1+Time_Real3 C1=bin(counts)[2:] C5=len(C1) C2=C5//8 C4=C5%8 if
C4!=0: C3=(C2+1)*8 else: C3=C2*8 C="0"+str(C3)+"b" Time_Real3=format(counts,C)
T1=len(Time_Real3) Time_Real1=bin(T1)[2:] T2=len(Time_Real1) Time_Real4=format(T2,'06b')
Counts=Time_Real4+Time_Real1+Time_Real3
```

extract

```
lenf6=len(File_information5) ascii_string = "" while File_information5[:8]!="00101111":
a_binary_string=File_information5[:8] binary_values = a_binary_string. split() for binary_value in
binary_values: an_integer = int(binary_value, 2) ascii_character = str(chr(an_integer)) ascii_string +=
ascii_character File_information5=File_information5[8:] File_information5=File_information5[8:]
while File_information5[:1]!="1": if File_information5[:1]=="0":
File_information5=File_information5[1:] File_information5=File_information5[1:]
#print(File_information5) #print(Extract_info) Real_C=int(File_information5[0:6],2)
File_information5=File_information5[6:] Real_C1=int(File_information5[:Real_C],2)
File_information5=File_information5[Real_C:] XR=int(File_information5[:Real_C1],2)
File_information5=File_information5[Real_C1:] Real_C=int(File_information5[0:6],2)
File_information5=File_information5[6:] Real_C1=int(File_information5[:Real_C],2)
File_information5=File_information5[Real_C:] Extract_info=int(File_information5[:Real_C1],2)
```

```
File_information5_17=format(Number_of_the_file,CN)
```

I want say that I started writing my compression 13 years ago. I made my first version Spring-1.0.0.0: 2 years ago it was. I use banachii encoding. I use algorithm in version 1.0.0.0 pi algorithm.

After in other versions I used reverse (change information) and compression it and after reverse it back.

In Spring-160 I used compression algorithm Spring.

Spring algorithm:

Use formula and it repeat many times:

Formula:

```
Number_of_the_file=((((Number_of_the_file*2**y)+Add)//3)*Multiplayer)//Divided_corrdiates
```

Number_of_the_file= for count numbers of the file

Divide_corrdiates.

T_Real it's about our reality where used to multiply numbers.

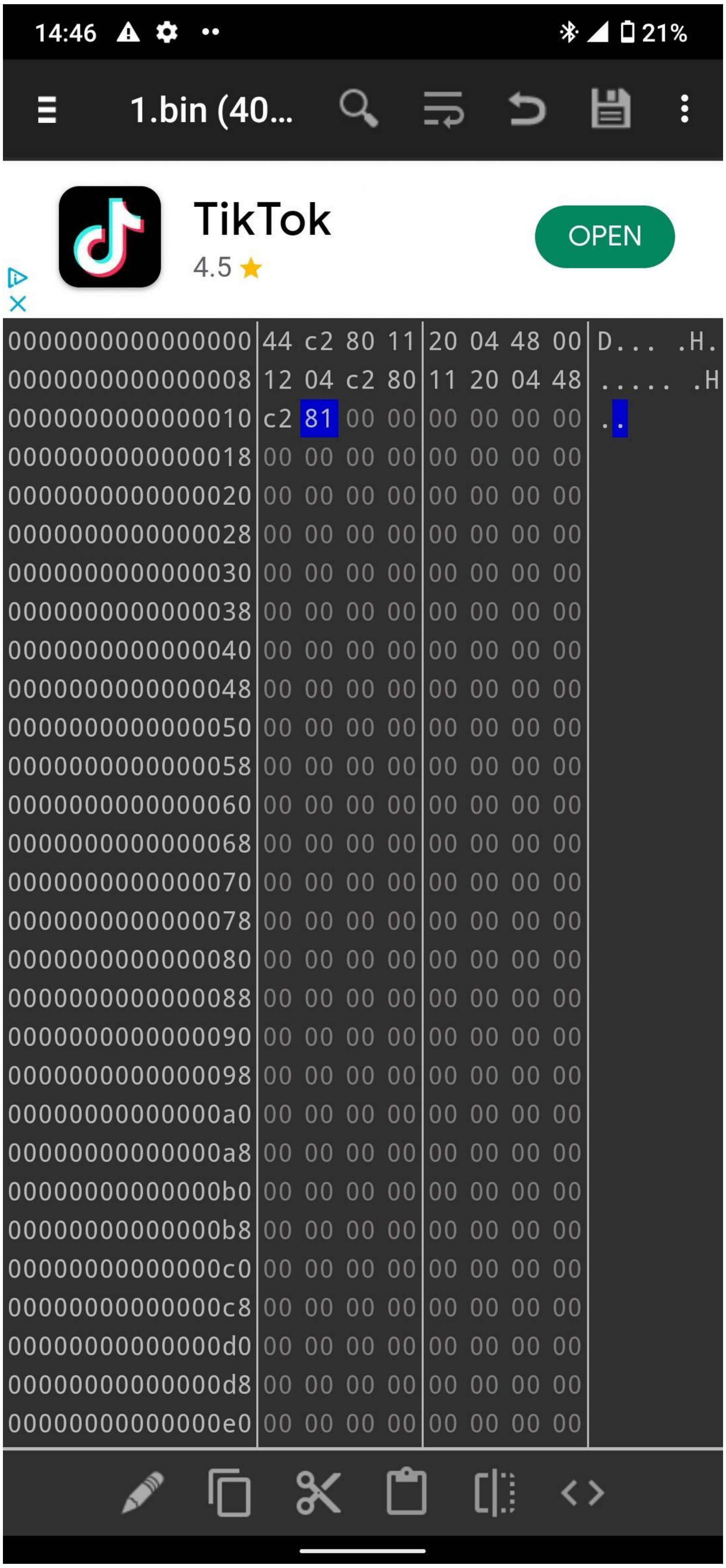
I have names it real because it let compression random data good question why need division it need for find little bit small number variation to compression

So how could compress by around numbers we need to use this formula try to get this number to out number we take all version and calculate it while we get our variation the shorted order so it's compressed by this formula shorter how we can with seven numbers. You saw why trace jumping like you said because that is banachii encoding. After we reverse it back.

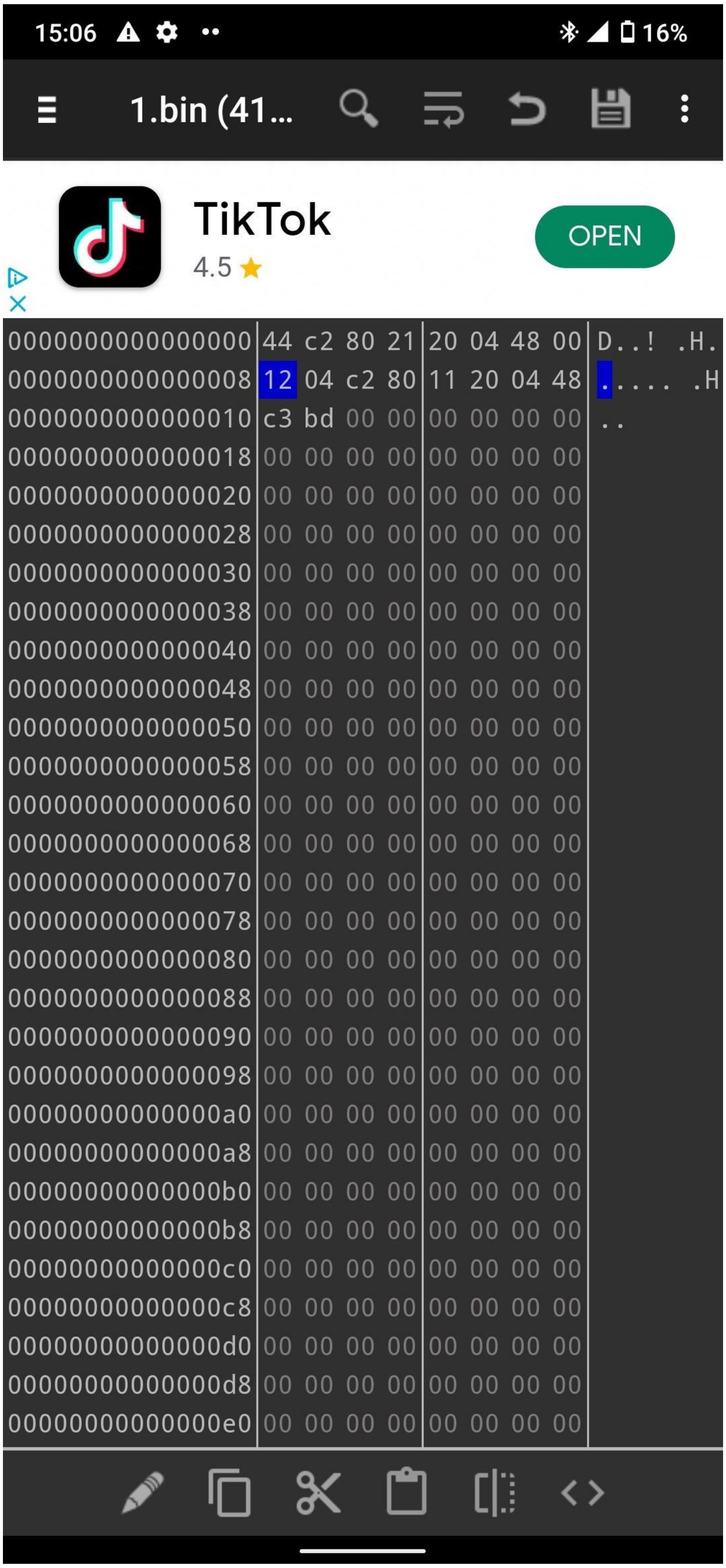
So I use this formula to extract by the same combination put this number that we get from formula end extract it back but after I use encoding banachii because it looks compression encoding to secure information after it extract back. Also I compress zeroes by used encoding zeroes and after save it. Like 40 bits. It's saving how many bits. I use banachii to encoding and decoding it.

Here is compression file

2b and 2) ff



```
0000000000000000 44 c2 80 11 20 04 48 00 D... .H.
0000000000000008 12 04 c2 80 11 20 04 48 ...... .H
0000000000000010 c2 81 00 00 00 00 00 00 ..
0000000000000018 00 00 00 00 00 00 00 00
0000000000000020 00 00 00 00 00 00 00 00
0000000000000028 00 00 00 00 00 00 00 00
0000000000000030 00 00 00 00 00 00 00 00
0000000000000038 00 00 00 00 00 00 00 00
0000000000000040 00 00 00 00 00 00 00 00
0000000000000048 00 00 00 00 00 00 00 00
0000000000000050 00 00 00 00 00 00 00 00
0000000000000058 00 00 00 00 00 00 00 00
0000000000000060 00 00 00 00 00 00 00 00
0000000000000068 00 00 00 00 00 00 00 00
0000000000000070 00 00 00 00 00 00 00 00
0000000000000078 00 00 00 00 00 00 00 00
0000000000000080 00 00 00 00 00 00 00 00
0000000000000088 00 00 00 00 00 00 00 00
0000000000000090 00 00 00 00 00 00 00 00
0000000000000098 00 00 00 00 00 00 00 00
00000000000000a0 00 00 00 00 00 00 00 00
00000000000000a8 00 00 00 00 00 00 00 00
00000000000000b0 00 00 00 00 00 00 00 00
00000000000000b8 00 00 00 00 00 00 00 00
00000000000000c0 00 00 00 00 00 00 00 00
00000000000000c8 00 00 00 00 00 00 00 00
00000000000000d0 00 00 00 00 00 00 00 00
00000000000000d8 00 00 00 00 00 00 00 00
00000000000000e0 00 00 00 00 00 00 00 00
```



```
0000000000000000 44 c2 80 21 20 04 48 00 D..! .H.
0000000000000008 12 04 c2 80 11 20 04 48 ..... .H
0000000000000010 c3 bd 00 00 00 00 00 00 ..
0000000000000018 00 00 00 00 00 00 00 00
0000000000000020 00 00 00 00 00 00 00 00
0000000000000028 00 00 00 00 00 00 00 00
0000000000000030 00 00 00 00 00 00 00 00
0000000000000038 00 00 00 00 00 00 00 00
0000000000000040 00 00 00 00 00 00 00 00
0000000000000048 00 00 00 00 00 00 00 00
0000000000000050 00 00 00 00 00 00 00 00
0000000000000058 00 00 00 00 00 00 00 00
0000000000000060 00 00 00 00 00 00 00 00
0000000000000068 00 00 00 00 00 00 00 00
0000000000000070 00 00 00 00 00 00 00 00
0000000000000078 00 00 00 00 20 00 48 00
0000000000000080 00 00 00 00 00 00 00 00
0000000000000088 00 00 00 00 00 00 00 00
0000000000000090 00 00 00 00 00 00 00 00
0000000000000098 00 00 00 00 00 00 00 00
00000000000000a0 00 00 00 00 00 00 00 00
00000000000000a8 00 00 00 00 00 00 00 00
00000000000000b0 00 00 00 00 00 00 00 00
00000000000000b8 00 00 00 00 00 00 00 00
00000000000000c0 00 00 00 00 00 00 00 00
00000000000000c8 00 00 00 00 00 00 00 00
00000000000000d0 00 00 00 00 00 00 00 00
00000000000000d8 00 00 00 00 00 00 00 00
00000000000000e0 00 00 00 00 00 00 00 00
```

It’s can compress random files and extract it back.