from base64 import b64decode as b64d from base64 import b64encode as b64e from random import randint import zlib # <------- Added library def flag\_generator(): flag = "CBA{Reverse\_engineering\_is\_awesome!}" #flag = flag.encode('zlib') # <------- This line causes an error # Convert flag string to binary binary\_flag = flag.encode() # Compress with additional zlib library flag = zlib.compress(binary\_flag) for i in range(randint(0, 10)): flag = b64e(flag) return b64e(flag) # The last step of the generator is base64 encoding a random number of times between 0-10 inclusive. # So create a loop that repeats 11 times. # Use a try block to attempt to 'decode' - the next required step. If it fails, keep looping, if not # end the loop early with a break, then # decode the file to reverse the zlib encoding on line 8 # print the flag def flag\_degenerator(flag): for i in range(11): flag = b64d(flag) try: #flag.decode('zlib') # Decompress with Zlib then convert from binary to text flag = zlib.decompress(flag).decode() break except: print ("Count no: " + str(i)) #flag = b64d(flag) #currentFlag = flag.decode('zlib') #print ("The flag is: " + currentFlag) print ("The flag is: " + flag) # Alternative solution - Using recursion to find the solution # Use a try block to test whether we can successfully decode the file. # If it can't, the function does a single base64 decode, then calls itself recusively until it can decode the file. # When it can decode the file, it does so and then prints, knowing that there are no more base64 decodes required. def flag\_degenerator\_recusion(flag): flag = b64d(flag) try: print ("The flag is: " + zlib.decompress(flag).decode()) except: flag\_degenerator\_recusion(flag) if \_\_name\_\_ == '\_\_main\_\_': #flag = "Vm0xd1MwNUdXWGxWV0dSUFZsZG9XRmx0ZUV0V01XeDBaVWRHVjFac1NsWlZWbEpIVlRKS1NHVk dXbFpOVmtwWVZrZDRTMk14V25GWApiSEJYVWxSV2VWZFdaRFJaVjAxNFZHNUdWQXBpUmxwWVdXd G9RMUpXV2toTlZGSmFWbTFTV0ZkcmFFOVZkM0JwVjBkb2RsWkdXbUZqCk1EQjRWMjVLWVZKRlNs QlZha0poVFVaYVNFNVZkRlZhTTBKWVdXdFdkMVZXV2xkaFNHUnFDazFXV25wV01uUmhWakpLY21 ORk9WZGkKV0dob1ZXcEdkMVpzV25SU2JGcFNWMFZLVmxaWE1UQmpiVlpYV2taV1ZXSnRVbkZEYX pGWFVtcFNWMDFxVmxSV1ZFcEhZMnMxVjFWcwpjRmNLVWxWd2IxWldVa2RXYlZaSFYyNVNVMkpGT lZkV01GWkxaR3hrVjFWclpGZGhlbFpZVld4b2MxZHRWblJsUmtwRVlrWmFXVlF3ClVuTlNSbkEy VFVSc1JGcDZNRGxEWnowOUNnPT0K" flag = flag\_generator() print ("Try to get the flag reversing the python script that generated it!") print ("Flag string: " + str(flag)) flag\_degenerator(flag) flag\_degenerator\_recusion(flag