Directions: Solve the following problems. All written work must be your own. See the course syllabus for detailed rules.

1. The Caesar cipher.

   (a) Encrypt the message "exchange all assets" using a Caesar cipher with a forward shift of 5 characters.

   (b) Decrypt the following message, which has been encoded with a Caesar cipher.

   `DPYLA OLTVU LFAVT VYYVD`

2. [JJJ 1.{9,10}.c] Let $d = \gcd(16261, 85652)$. Use the extended Euclidean algorithm to find integers u and v such that $16261u + 85652v = d$.

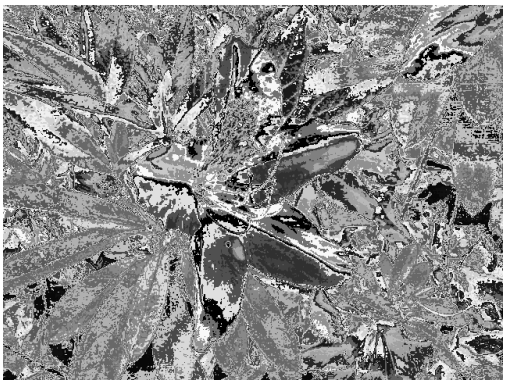
3. [**To be submitted with HW2 on Jan 26, 2022.**] A grayscale image with depth d is a two-dimensional array whose entries are in $\{0, \ldots, d-1\}$. For example, the following image T has depth 4.

$$T = \begin{bmatrix} 0 & 0 & 2 & 1 \\ 0 & 0 & 2 & 3 \\ 3 & 3 & 2 & 1 \end{bmatrix}$$

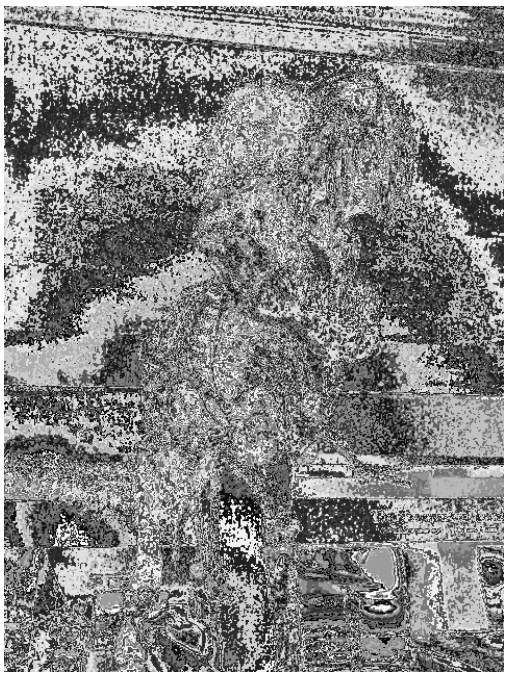
plaintext	0	1	2	3
cyphertext	3	1	0	2

$$C = \begin{bmatrix} 3 & 3 & 0 & 1 \\ 3 & 3 & 0 & 2 \\ 2 & 2 & 0 & 1 \end{bmatrix}$$

   When the plaintext image T is encrypted with a substitution cypher whose key is displayed in the center table, the result is the cyphertext image C, displayed on the right. The images `A.png` (depth 32), `B.png` (depth 64), `C.png` (depth 128), `D.png` (depth 256) contain grayscale images that have been encrypted with a simple substitution cypher; the keys are unknown.

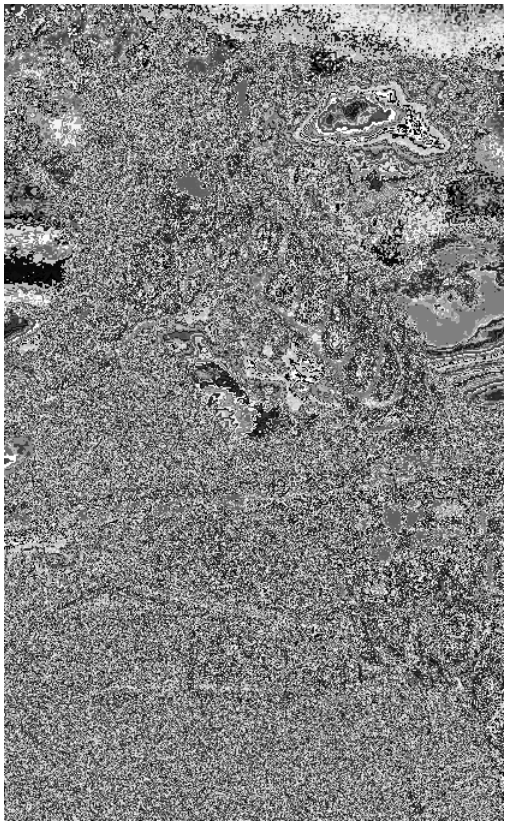


`A.png` (32 colors, scaled)



`B.png` (64 colors, scaled)

`C.png` (128 colors, scaled)



`D.png` (256 colors, scaled)

(a) Find a formula for the size of the keyspace for a substitution cypher on an image with depth d. If a computer can check 1 million keys per second, how many years would it take to decrypt `A.png` and `D.png`?

(b) Write some efficient code that breaks this cryptosystem, and use your code to decrypt the given images without knowledge of the key. The python template file `im_sub_cypher.py` contains some routines used to encrypt, decrypt, and save image files. To use the template, you will need the common python packages numpy and the Python Image Library. Note: you will probably need to test and debug your code on images that you encrypt and for which you know the encryption key. Hint: when designing your algorithm, exploit that nearby pixels are likely to have similar brightness levels.

(c) Describe in English how your algorithm works.

(d) Give the 4 plaintext images that your algorithm produces when running on the given files `A.png` (depth 32), `B.png` (depth 64), `C.png` (depth 128), and `D.png` (depth 256).