Try to code the assignment by yourself. Plagiarism is not tolerated

Assignment 2
Image Generation

Problem Statement

In this assignment you have to implement an image generator using mathematical functions. Read the instructions for each step. Use python with the **numpy** library.

Your program must allow the user to provide parameters in order to generate images by the following steps:

1. **Parameter input:**
   a. filename for the reference image r
   b. lateral size of the scene C (the scene is assumed to be square so that its size is C × C)
   c. the function to be used f (1, 2, 3, 4 or 5)
   d. parameter Q
   e. lateral size of the digital image N (also forming a square so that the size is N × N), and N ≤ C
   f. number of bits per pixel B, with 1 ≤ B ≤ 8
   g. seed S to be used for the random function
2. **Generate scene image**, f, according to the selected function and parameters.
3. **Generate digital image**, g, with sampling and quantisation defined by N and B.
4. **Compare** g, with the reference image r.
5. **Print** in the screen the root mean squared error between g and r.

Scene image, digital image

Scene image: functions to generate images

1. $f(x, y) = (xy + 2y)$
2. $f(x, y) = |\,cos(\frac{x}{Q}) + 2\,sin(\frac{y}{Q})|$
3. $f(x, y) = |3(\frac{x}{Q}) - \sqrt[3]{\frac{y}{Q}}|$
4. $f(x, y) = rand(0, 1, S)$

   The random function is uniform betwen 0 and 1, using seed S initialised once before the first number is sampled. Use random.random() for this function.

5. $f(x, y) = randomwalk(S)$

   Seed S is initialized once before the first number is sampled. Then, consider $f(x, y) = 0$ for all x, y. The random walk starts by setting the value 1 to the position $(x = 0, y = 0)$, i.e.$f(0, 0) = 1$. Then, random steps are computed considering at the same time x and y, generating a random number d_x between −1 and 1 and a random number d_y also between −1 and 1. Use **random.randint()** in this case. The program then sets $x = [(x + dx) \, mod \, C]$, $y = [(y + dy) \, mod \, C]$ and finally $f(x, y) = 1$. The module operator is important to avoid error of beyond matrix limits.

   The total number of steps (a step is given after each d_x and d_y sampling) is $1 + (C \cdot C)$

Use the package **random**; The scene image f must be computed using float type values. After f is computed, normalize values so that the minimum is 0 and the maximum is $2^{16} - 1 = 65535$

Sampling and quantisation steps: in this part, we simulate "digitizing" the image, generating an integer matrix g with size N × N and storing pixels with a maximum value of B bits (B between 1 and 8). Because g may have lower resolution than f a downsampling pooling operator must be employed. For example, consider a matrix g with C = 4.

$$\mathbf{g} = \begin{bmatrix} 5 & 15 & 36 & 0 \\ 18 & 0 & 0 & 1 \\ 0 & 100 & 154 & 0 \\ 0 & 99 & 159 & 100 \end{bmatrix}$$

This downsampling operator takes the first pixel in a given region and skips the remaining ones. For an image f with N = 2 we would have:

$$\mathbf{f} = \begin{bmatrix} 5 & 36 \\ 0 & 154 \end{bmatrix}$$

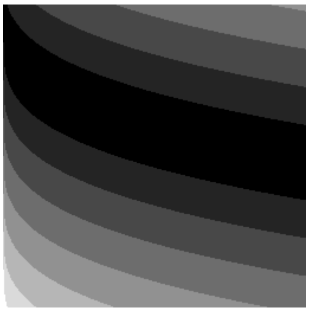
The step can be defined as the integer ratio between C and N, i.e. $\lfloor C/N \rfloor$.

Note that $g(0, 0) = f(0, 0)$ and then $g(0, 1)$ is obtained by skipping a number of pixels relative to the ratio of reduction between f and g.

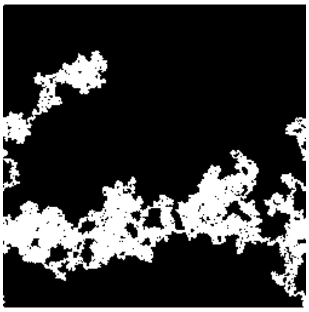
In addition, f may contain values higher than 2^8. Thus, a quantisation is needed, using a bitwise shift. In order to perform that, first convert values of f into a 8-bit unsigned integer, so that the maximum value is $(2^8) - 1 = 255$. Then, perform a bit-shift so that only the B most significant bits remain, and the other ones are only zeros.

Examples of figures generated by the 5 different functions can be seen below:

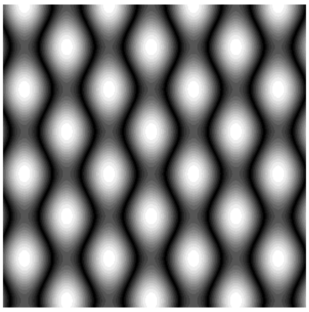
1



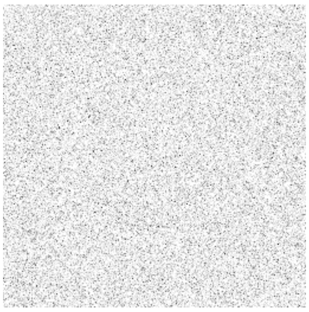
3 (Q = 1001, B = 3)



5 (S = 6666, B = 8)



2 (Q = 32, B = 6)



4 (S = 13, B = 3)

Comparing with reference

Your program must compare the generated image with a reference image r. This comparison must use the root squared error (RSE). Print this error in the screen, rounding to 4 decimal places.

$$\text{RSE} = \sqrt{\sum_i \sum_j (g(i,j) \; - \; R(i,j))^2}$$

Note this formula does not divide the error by the number of pixels. It is a modification of the Root Mean Squared Error, showing the sum of the errors in all pixels.

The reference image is stored in the form of a numpy matrix. You should load and convert to the uint8 to assure the comparison is valid, as below:

Example

```
import numpy as np
filename = str(input()).rstrip()
R = np.load(filename)
```

Input and Output

Example of input:

Reference image in the file ex1.npy, C = 1024, function 1, parameters: Q = 2, N = 720, B = 6, S = 1

Input

```
ex1.npy
1024
1
2
720
6
1
```

Note function 1 does not use parameters Q and S, still all must be read via keyboard.

Example of output:

Only the RSE value in float format.

Example 1 (high RSE, indicating the generated image is too different from the reference):
Example 2 (lower RSE, indicating a similar image and a correct result):

Output

```
7468.7864
```

Submission

Submit your source code using the Run.Codes (only the .py file)

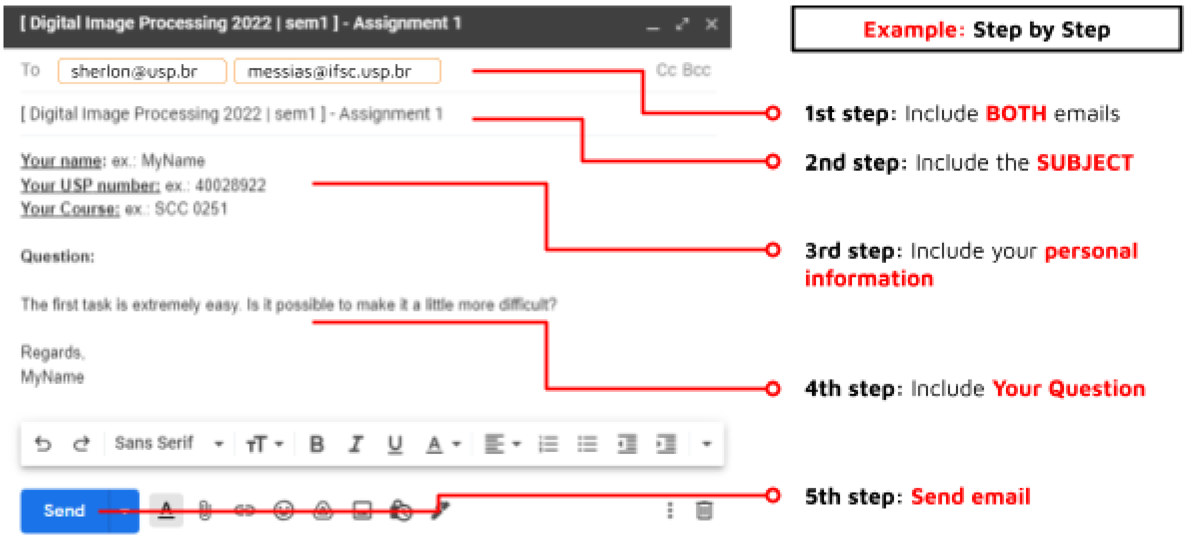
1. **Comment your code.** Use a header with name, USP number, course code, year/semester and the title of the assignment. A penalty on the grading will be applied if your code is missing the header and comments.
2. **Organize your code in programming functions.** Use one function per method.

Contact

If you have any questions, contact us by sending an email following the five steps below:

1st step: Include **BOTH** emails, sherlon@usp.br **and** messias@ifsc.usp.br.
2nd step: Include the subject **exactly** like this:
Subject: "**[Digital Image Processing 2022 | sem1] - Assignment 1**"
Do not change the initial part (**black**).
Replace the final part with the topic you are interested in (**red**).
3rd step: Add your personal information to help us find your submissions in Run.Codes and E-Disciplinas quickly.
4th step: Formulate your question in detail. Include your implementation and/or screenshots if necessary.
5th step: Send email and wait. We will respond as soon as possible.

Example of Email:



[Digital Image Processing 2022 | sem1] - Assignment 1

To sherlon@usp.br messias@ifsc.usp.br Cc Bcc

[Digital Image Processing 2022 | sem1] - Assignment 1

Your name: ex.: MyName
Your USP number: ex.: 40028922
Your Course: ex.: SCC 0251

Question:

The first task is extremely easy. Is it possible to make it a little more difficult?

Regards,
MyName

Example: Step by Step

1st step: Include **BOTH** emails

2nd step: Include the **SUBJECT**

3rd step: Include your **personal information**

4th step: Include **Your Question**

5th step: **Send email**