# Image Median Filter Parallelization – Coding Challenge

## Description of the Challenge

Local median filter is used for smoothing images and noise reduction. The degree of smoothing is determined by the kernel size which can be 2D or 3D. Applying 3D median filter to large 3D images (eg. 2000x2000x2000 voxels) is challenging because the images often do not fit in the memory and single-threaded execution of the filter is inefficient and too slow.

## Challenge 1- Median Filter Parallelization.

Feel free to apply the full extent of your expertise to approach this problem using python, C and Java. As a suggestion, parallelization can be done by splitting the image into sub volumes (tiling/gridding method):

* Create a 3D synthetic image and split it into subvolumes and apply median filter (like skimage.filters.median) to each subvolume in parallel. The output needs to be a 3D array with the same size as the original 3D image.
* How does the performance scale for a fixed kernel size as a function of subvolume size and number of threads.
* Explain the results of the benchmark. Are there any performance bottlenecks? Is this method accurate? Can you quantify the error? What are your suggestions for improving this method?

## Challenge 2- Quantitatively compare the method you developed in Challenge 1 to existing methods like Dask-image or GPU-based parallelization like Cucim where applicable.

## Evaluation

Please make your contribution available to people with basic python and image processing skills. Your contribution primarily will be evaluated based on the clarity of your code, ideas and explanations.

## Document format

Jupyter notebook is preferred.