Mandelbrot Set – Part 3

Numerical Scientific Computing Mini-Project

Name: Lukas Bisgaard Kristensen

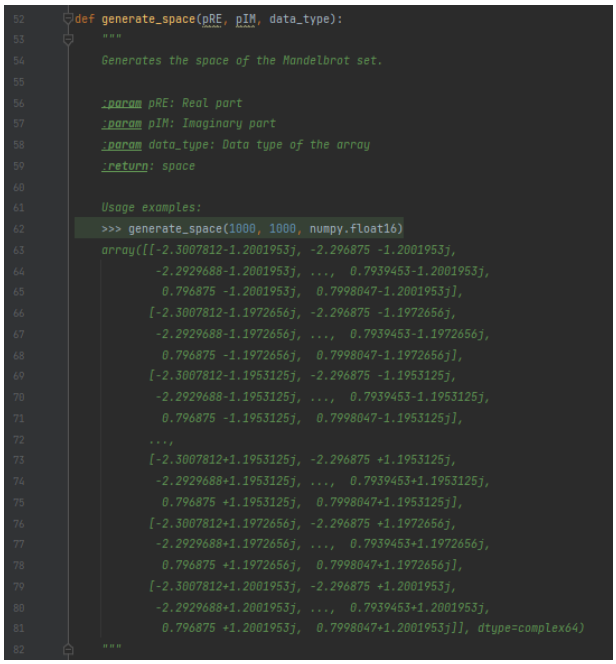
Date: 26/04/2023

Program: Computer Engineering (AVS), 8th semester, Aalborg University

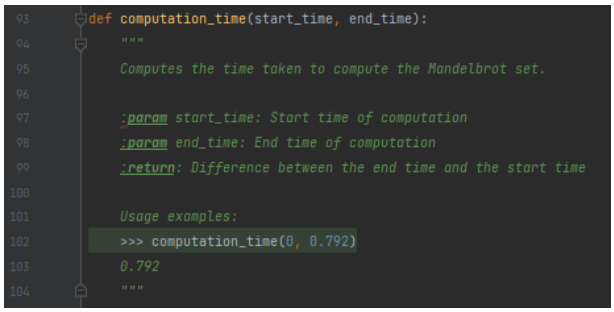
Course: Numerical Scientific Computing

Docstrings and Doctests (3 cases), see mandelbrot_vectorized.py

```
def mandelbrot(c):
    """
    Computes the Mandelbrot set using vectorized numpy operations.

    :param c: Input complex array
    :return mandelbrot: Divergence time

    Usage examples:
    >>> mandelbrot(numpy.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]) * 1j)
    array([0., 0., 1., 0., 0., 0., 0., 0., 0., 0.], dtype=float16)
    >>> mandelbrot(numpy.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]) + numpy.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]) * 1j)
    array([0., 1., 0., 0., 0., 0., 0., 0., 0., 0.], dtype=float16)
    """
```



```
def generate_space(pRE, pIM, data_type):
    """
    Generates the space of the Mandelbrot set.

    :param pRE: Real part
    :param pIM: Imaginary part
    :param data_type: Data type of the array
    :return: space

    Usage examples:
    >>> generate_space(1000, 1000, numpy.float16)
    array([[-2.3007812-1.2001953j, -2.296875 -1.2001953j,
            -2.2929688-1.2001953j, ...,  0.7939453-1.2001953j,
             0.796875 -1.2001953j,  0.7998047-1.2001953j],
           [-2.3007812-1.1972656j, -2.296875 -1.1972656j,
            -2.2929688-1.1972656j, ...,  0.7939453-1.1972656j,
             0.796875 -1.1972656j,  0.7998047-1.1972656j],
           [-2.3007812-1.1953125j, -2.296875 -1.1953125j,
            -2.2929688-1.1953125j, ...,  0.7939453-1.1953125j,
             0.796875 -1.1953125j,  0.7998047-1.1953125j],
           ...,
           [-2.3007812+1.1953125j, -2.296875 +1.1953125j,
            -2.2929688+1.1953125j, ...,  0.7939453+1.1953125j,
             0.796875 +1.1953125j,  0.7998047+1.1953125j],
           [-2.3007812+1.1972656j, -2.296875 +1.1972656j,
            -2.2929688+1.1972656j, ...,  0.7939453+1.1972656j,
             0.796875 +1.1972656j,  0.7998047+1.1972656j],
           [-2.3007812+1.2001953j, -2.296875 +1.2001953j,
            -2.2929688+1.2001953j, ...,  0.7939453+1.2001953j,
             0.796875 +1.2001953j,  0.7998047+1.2001953j]], dtype=complex64)
    """
```



```
def computation_time(start_time, end_time):
    """
    Computes the time taken to compute the Mandelbrot set.

    :param start_time: Start time of computation
    :param end_time: End time of computation
    :return: Difference between the end time and the start time

    Usage examples:
    >>> computation_time(0, 0.792)
    0.792
    """
```

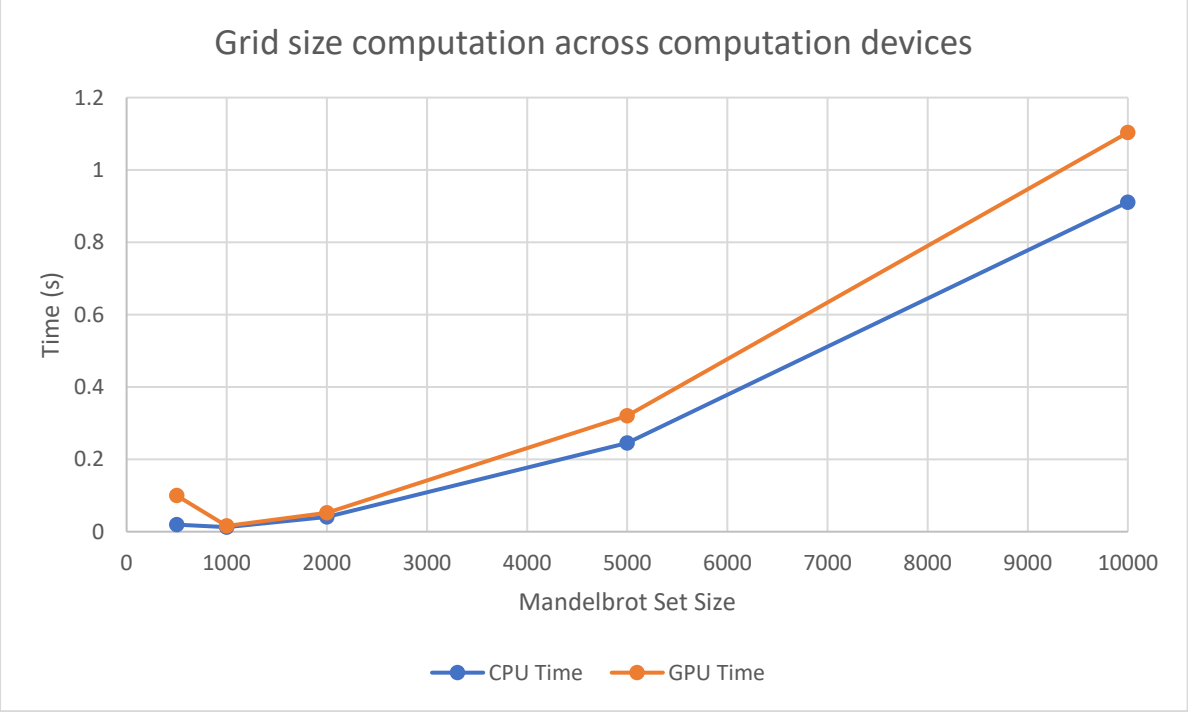
OpenCL with defined memory types for all variables

__global memory data type for the input and output data.

__private memory data type for data that is only relevant for workers within the function.

```
__kernel void mandelbrot(__global float2 *complete_space, __global float *output)
{
    __private int gid = get_global_id(0);
    __private float nreal, real = 0;
    __private float imaginary = 0;
    __private float real_pow_2, imaginary_pow_2;

    for (int i = 0; i < 100; i++)
    {
        real_pow_2 = real * real;
        imaginary_pow_2 = imaginary * imaginary;

        nreal = real_pow_2 - imaginary_pow_2 + complete_space[gid].x;
        imaginary = 2 * real * imaginary + complete_space[gid].y;
        real = nreal;
        if (real_pow_2 + imaginary_pow_2 > 4)
        {
            output[gid] = i;
            return;
        }
    }
}
""").build()
```

Benchmarking results for grid-sizes across computation parameters



Extra features

- Zoom Animation
    - Code: mandelbrot_iteration_animation.py
        - GitHub: https://github.com/LukasKristensen/Mandelbrot-Python/blob/main/Iterations%20Animation/mandelbrot_iteration_animation.py
    - Video of output: https://www.youtube.com/watch?v=L2zKIrriDfI

- Iteration Animation
    - Code: mandelbrot_animation.py
        - GitHub: https://github.com/LukasKristensen/Mandelbrot-Python/blob/main/Zoom%20Animation/mandelbrot_animation.py
    - Video of output: https://www.youtube.com/watch?v=8BjqgaIuses