This planning report outlines the intended structure and implementation of assignment 2. The task is to implement a program, in the programming language C, that iterates Newtons formula applied to a polynomial equation of the form $X^d - 1 = 0, d \leq 10$. The computation is carried out of for each cell of the $L \times L$ uniform grid that covers $2 + 2i$ to $-2 - 2i$ on the complex plane. The result of the program is two images, where each pixel corresponds to one of these grid cells. In the first image the cells are coloured according to the iteration steeps needed to converge to a root. In the second case the pixels are coloured according to which of the $d + 1$ roots the iteration converges($d + 1$ to accommodate divergence and points close to the origin.) Additionally the program should be able to split the work over T threads as specified by the user.

1 Relevant concepts

Below follows a list of concepts that are deemed relevant to the implementation of the program.

1. **Memory allocation** - As with most C programs a thorough understanding of memory and it's allocation is required. This includes a proper use of dynamically allocated memory (heap memory allocated by calls to the function malloc), local variables which are placed on the stack and statically allocated global variables. Improper use of these may result underperforming program. For example overzealous application of global variables will make it hard for the compiler to analyse and optimise the program structure.

2. **Reading assembler** - When writing performance critical code it can be very useful to be able to read and to some extent understand the compiler generated assembler. This removes some of the guess work in the process, by for example detect if the compiler inlines a certain math function or uses an actual function call.

3. **Threading and paralization** - As the program should utilise pthreads it is essential to understand threads and the synchronisation their synchronization mechanisms. In addition we intend to explore the gcc extension atomic as an plausible alternative to a mutex as described in the lectures.

4. **Vectorization** - Understanding the CPU's ability to, in parallel, work on multiple data using vector instructions. Whilst not the primary focus of the assignment they can probably provide some performance boost. Vectorization can either be applied manually by the use of vector intrinsic functions or automatically by the compiler if the code structure allows for it. In the latter case it would be useful to inspect the assembly for verification.

5. **Argument passing & parsing** - The parameters T,d and L are to be provided by the program user from the command line. Thus utilisation of POSIX or C standard library argument parser function might be of use. Alternatively could the code from assignment 0 be repurposed.

6. **Efficient file writing** - The image format employed in this assignment is text based (as opposed to binary) thus an efficient strategy for writing these files are needed. The lessons of the previous assignments should probably apply. For example we know that writing blocks directly is much faster than using the function fprintf or writing bytes individually in a for loop.

2 Program structure

The program is split into three main part (which preferably are separate compilations units) as follows.

1. **Writer thread** - A function that runs on a separate thread and tasked with writing the data to file.

2. **Worker thread** - A function that runs on T different threads in parallel.

3. **Manager** - The manager parses the command line, manages memory allocation and initialises all the the needed threads.

2.1 Writer thread details

The writer writes each line as is is calculated to a file. It's implementation can further be divided into several sub tasks.

- **File header** - The first task of the worker is to write the file header of the PPM file as outlined in the file description.

- **Pixel calculation** - Each grid cell computation yields two values, number of iterations and root converged to. These values needs to be converted from binary integer format to a text string of the for "RR BB GG" representing the pixels colour. For the limited number of roots and grey levels required in this assignment a precomputed lookup table can be employed.

- **Synchronisation** - The worker needs to wait for a row to be ready, and first then process the row. This is easily accomplished using a flag array as shown in the lecture.

- **File writing** - Once a rows data has been received and converted into an text string it should be written to the file.

2.2 Worker thread

The worker thread shall process the computation in a striped fashion to equalise work as described in the lecture. I.e thread 0 processes row 0,T,2T,... whilst thread Thread T processes rows T-1,2T-1,... For each element of the row the worker has two main problems to solve.

1. Efficiently evaluate the newton iteration step
2. Efficiently evaluate the stopping condition, and if it is met halt the iteration store the result.

Once the row is completed the writer thread should be notified.