1. **Name, Student ID:**
    - Name: 葉之晴
    - StudentID: 110065504

2. **Explain your implementation, especially in the following aspects:**
    - How do you implement your program, what scheduling algorithm did you use: static, dynamic, guided, etc.?

      First, calculate the color of the row and column and bring in i,j line to calculate the color. Second, consider the block splitting. The task is split into row units. Distribute row evenly to each core, which then turns on MP. In order to avoid numerical errors, lock it up before the program finishes calculating the row.

    - How do you partition the task?

      The task is split into row units.

    - What techniques do you use to reduce execution time?

      MPI and pthread have been tried separately, and both have a certain improvement. But it may be that there is something wrong in the allocation of the resource so that both of them couldn't be used at the same time, so I ended up using MPI, MP, and optimization of some of the code.

      A detailed description of three tried edition code

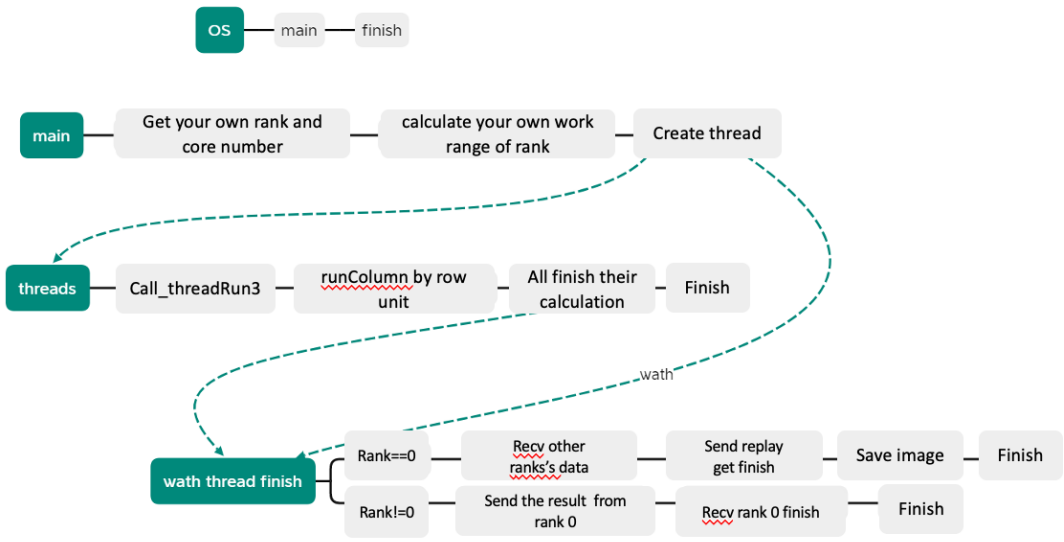
      A. threadRun: The first version, after creating with thread, come in and take only one row at a time to calculate.
      B. __threadRun: MPI and MP version, after calculating the number of threads that can be used, the number of operations is evenly distributed to each threads.
      C. __threadRun3 : MPI, MP ,and pthread version, which is a combination of 1 and 2. After MPI, the thread can't be opened, so it's useless.

- Other efforts you make in your program.

  Calculation of the image margin to make the program end early.

  A. Count the right side, assume that if more than five are black, it will not be processed later. This assumption is correct after I tried.
  B. Count the left side, assume this value can be passed perfectly, otherwise some cannot pass like task6. The best value is 120 after I tried.

3. **Analysis:**
   - Design your own plots to show the load balance of your algorithm between threads/processes.



4. **Conclusion:**
   - What have you learned from this assignment?

     When it comes to rendering, the Mandelbulb can be colored just like any other 3D fractal. So the improvements I made mostly focused on rendering, to detect the color distribution

of the task. In the division of tasks, the unit is a row, and they are assigned to threads to complete.

Because there was something that happened at home this week, I was in a hurry for hw2. I found that this kind of problem is very common in parallel programs in the past few days, and other people optimized the shape of the segmentation problem can bring a good improvement. Maybe I'll try more later. Also, next time, I will remember to read more information and then optimize the program again. I hope that I can improve myself more from other people's programs.

- What difficulty did you encounter in this assignment?

  In this assignment, in addition to being familiar with the parallel computing that the professor taught in class, I paid more attention to understanding the actions related to computer graphics. I needed to know how it works so I could split the task further.

  Also, after I built MPI, I wanted to build pthread to speed up. But the program showed errors all the time and I couldn't solve this problem. Therefore, I only used MPI, MP, and a little algorithm to optimize. This is something trouble I ran into.

- Any feedback or suggestions to this assignment or spec.

  Excuse me, can you provide the code of those who are not a student in this class with high marks after the homework is over? So maybe we can learn from them. Thanks a lot.

  Also, I hope the TA can tell me what method I am missing because I postponed the assignment for one day to adjust the parameters and adjust the shape of the partition task, and I was only improved by 30 seconds, which made me very disappointed.