NTNU
Norwegian University of
Science and Technology

Department of Computer Science

TDT4200 - Parallel Programming

Exercises 0 (Optional)

1 Introduction

This is an entirely voluntary exercise and is meant as an introduction to the programming language `C`. Completing this exercise is recommended but does not grant you any marks towards being able to take the final exam. If you find this exercise challenging, it is recommended that you familiarise yourself further with the language. For example, implement `Game of Life`.

2 Tasks

2.1 Programming

In this assignment, you will alter a bitmap image. A skeleton code is available on Blackboard. This code contains functions to save and load a bitmap image. You will load a provided image, Figure 1 and make changes. How you alter the image is up to you, but some possibilities are inverting the colours, flipping the image, or remapping the colours. The final image should be recognisable. You will also resize the image. Simple interpolation (duplicating pixels) is sufficient for filling the new pixels. You must also create an automated method of compiling the code using a `Makefile`. Remember to comment on your code!

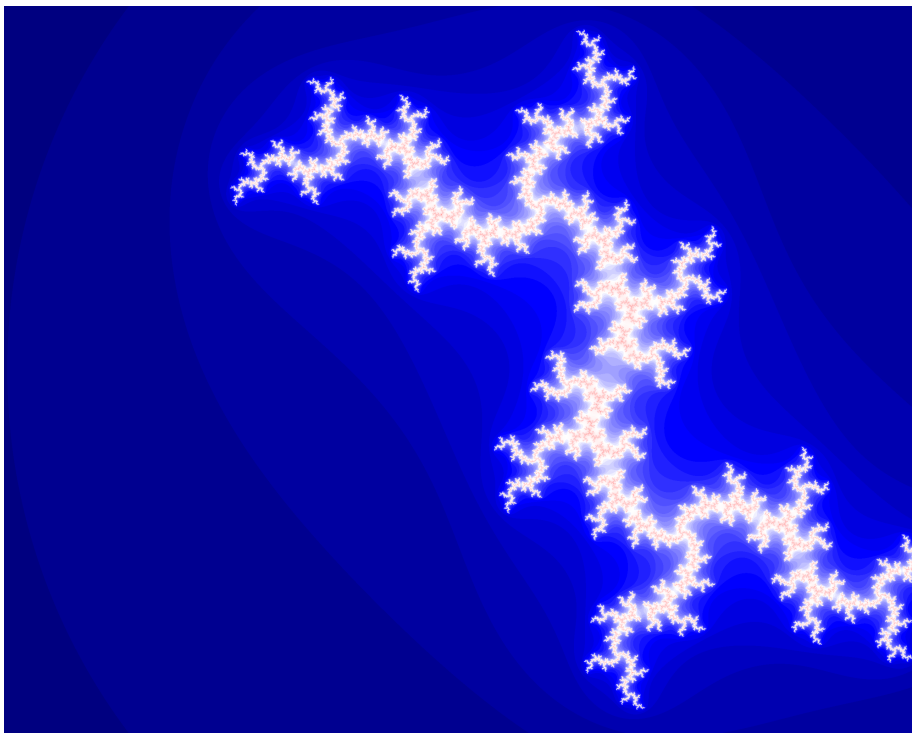

Figure 1: The provided "before.bmp" file.

Notice the following line in `main.c`

```
1 uchar *image = calloc(XSIZE * YSIZE * 3, 1);
```

This means the reserved space for the `.bmp` image is a single consecutive array. Each pixel in the `.bmp` image consists of 3 colour channels (RGB), as shown in Figure 2, where each colour channel is an `unsigned char` (1 Byte).

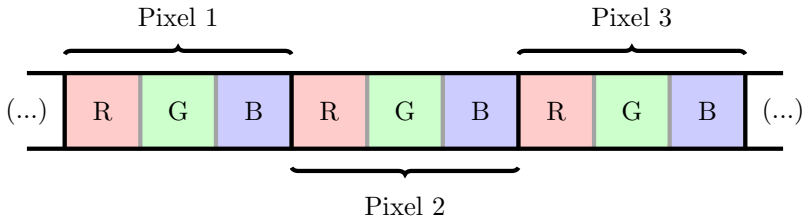

Figure 2: Illustration of how the `.bmp` is stored.

Requirements:

- Add at least one method (procedure or function) to the program that alters the starting image.
- Resize the image to double the image size (i.e. double the resolution in each direction).
- Change the image's colour to your choice.
- Your delivery must contain a `Makefile` to produce the executable named "bitmap".
- This executable should produce an image named "after.bmp" when run.
- The work must be done in the language `C`.

The results from this exercise may vary, but an example is shown in Figure 3.

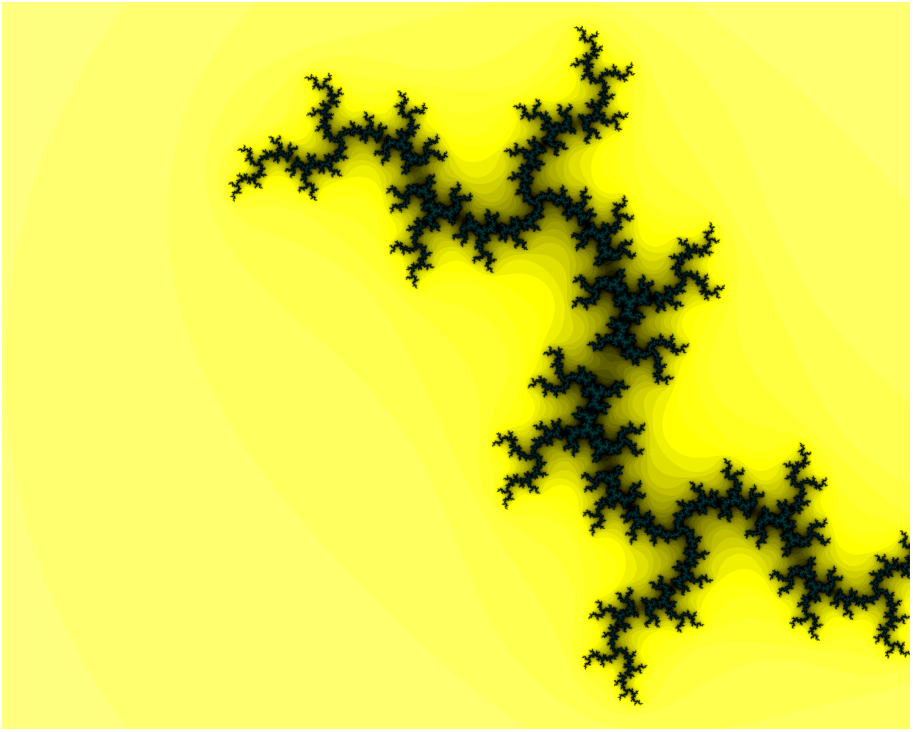

Figure 3: "after.bmp" with resize and a change of colour.

2.2 Theory questions

1. What is the purpose of a `Makefile`?
2. What is a pointer in `C`?

3. What does the `-O3` flag do when compiling `C` code?
4. How can you pass a value by reference to a function in `C`?

3 Deliverables

Please deliver `.zip` file with the following contents containing your changes to the code:

- `bitmap.h`
- `bitmap.c`
- `main.c`
- `Makefile`
- `before.bmp`
- A document with the answers to the theory questions.