C Programming II
2022 Spring
Homework 03

Instructor: Po-Wen Chi

Due: 2022.04.12 PM 11:59

Policies:

- **Zero tolerance** for late submission.
- **Plagiarism is not allowed.** Both source and copycat will be **zero**.
- You need to prepare a README file about how to make and run your program. Moreover, you need to provide your name and your student ID in the README file.
  - Your Name and Your ID.
  - The functional description for each code.
  - Anything special.
- Please pack all your submissions in one zip file. **RAR is not allowed!!**
- For convenience, your executable programs must be named following the rule hwXXYY, where the red part is the homework number and the blue part is the problem number. For example, hw0102 is the executable program for homework #1 problem 2.
- I only accept **PDF**. MS Word is not allowed.
- Do not forget your Makefile. For convenience, each assignment needs only one Makefile.

1 Bible

The Bible is a collection of religious texts, writings, or scriptures sacred in Christianity. It is is widely considered to be the best-selling book of all time. This time, I want you to develop a search function for the English Bible. I provide you a text file, **bible.txt**, which contains all verses. Your program should work as follows:

```
$ ./hw0301
Please enter the search target: in the beginning
Found 1 time(s)
1. Gen 1:1 In the beginning God created the heavens and the earth.
```

Note that the above is just an example and I do not guarantee the search correctness.

2 SRT Player (20 pts)

The SubRip file format is described on the Matroska multimedia container format website as ”perhaps the most basic of all subtitle formats.” SubRip (SubRip Text) files are named with the extension **.srt**, and contain formatted lines of plain text in groups separated by a blank line. Subtitles are numbered sequentially, starting at 1. The timecode format used is hours:minutes:seconds,milliseconds with time units fixed to two zero-padded digits and fractions fixed to three zero-padded digits (00:00:00,000). The fractional separator used is the comma, since the program was written in France.

This time, I want you to develop a SRT player.

```
$ ./hw0302
Open a srt file: JamieOliver_2010-480p.eng.srt
Speed (0.5-2): 1
```

Then, clear the screen with **system(”clear”);**. From 00:00:12,760 to 00:00:14,736, the screen will be

```
Sadly,
```

And from 00:00:14,760 to 00:00:18,023, the screen will be

```
in the next 18 minutes when I do our chat,
```

The speed is used to modify the subtitle player's speed. If it is set to 0.5, then 1 second appearing time will be extended to 2 seconds; if it is set to 2, then 1 second appearing time will be decreased to 0.5 second. For your convenience, I promise the input file is a valid srt file.

3 Circular Focus (20 pts)

Given a BMP picture, please generate a picture with a circular focus, as shown in figure 1. You should fill the redundant part with white color and you do not need to draw boundaries.

You need to make user input the center of circle (x, y) first and then enter the radius r. The coordinate system is shown in figure 2. For your simplicity, x, y, r are guaranteed to be positive integers.

Note that you need to handle the over boundary case, as shown in 3.

```
$ ./hw0303
Please enter the input image name: doraemon.bmp
Please enter the output image name: output.bmp
Please enter the center: (200,480)
Please enter the radius: 100
```

Note that the precision is not a concern in this problem. Note that the user may input a file which is not a BMP file, you need to give a warning. How to check if the input is a BMP file? Read the first two bytes.

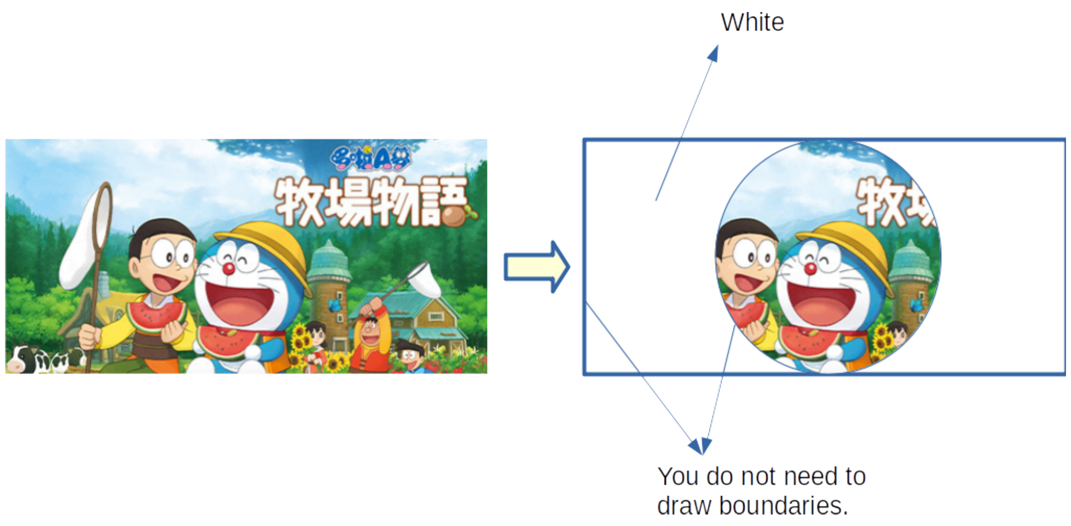

Figure 1: Circular focus.

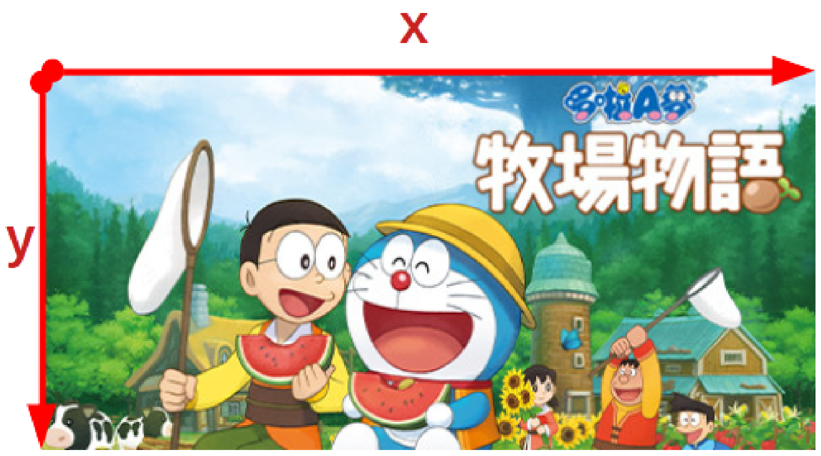

Figure 2: Coordinate system.

4 BMP with 16 bits (20 pts)

Digital images consist out of pixels. If we want to store or transmit an image, we have to provide information about each individual pixel. A common representation of color information is to use 3 bytes to represent RED, GREEN, BLUE respectively. This is what I have shown you in this class. However, when I was young, we did not use 3 bytes to present a pixel. Instead, we preferred to use **2 bytes** to present one pixel. So there are only 65535 possible colors. How to represent 3 colors in only 2 bytes? We use 5 bits for Red and Blue and 6 bits for Green. The transformation is presented in figure 4. Note that since we have less bits (information) available, we can represent less colors.

For your reference, I provide you BMP wikipedia and two BMP files, one is 3-bytes presentation and the other is 2-bytes presentation. Your program should be

```
$ ./hw0304
Please enter the input image name: maldives.bmp
Please enter the output image name: maldives_16.bmp
```

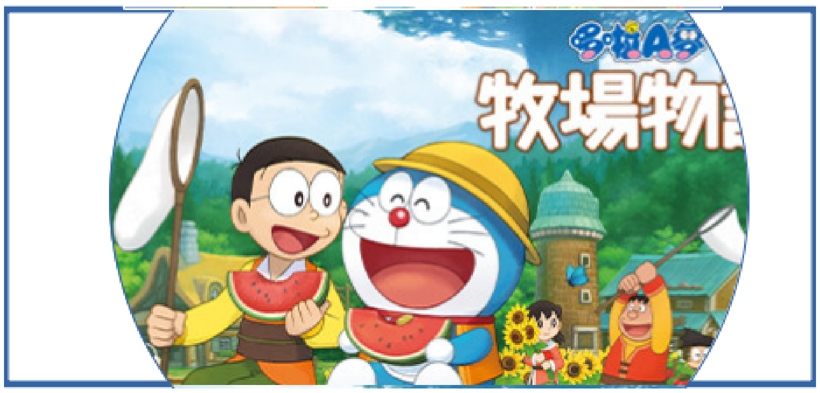

Figure 3: Remember to handle the over boundary case.

```
Done!
```

Note that the user may input a file which is not a BMP file, you need to give a warning. How to check if the input is a BMP file? Read the first two bytes.

For your reference, I show you how I generate the 16 bits file. I use a tool called **gimp**, a very famous open source image processing tool, ans export the figure as shown in Fig. 5.

5 Sliding Puzzle (20 pts)

A sliding puzzle, sliding block puzzle, or sliding tile puzzle is a combination puzzle that challenges a player to slide (frequently flat) pieces along certain routes (usually on a board) to establish a certain end-configuration. Please see the wikipedia for reference.

`https://en.wikipedia.org/wiki/Sliding_puzzle`

Do not worry, I do not ask you to make a sliding puzzle solver. In stead, I want you to develop a tool to create a sliding puzzle. First, the user need to input a BMP file as the puzzle background and a puzzle generation text file. The text file format is as follows:

```
3 3
2
1 2 3
4 5 0
7 8 9
1 2 3
4 0 5
7 8 9
```

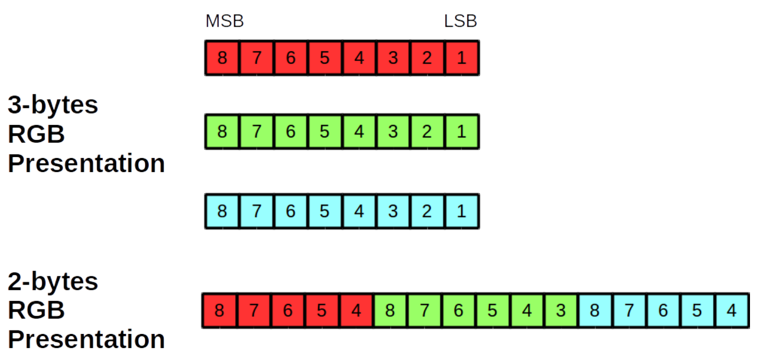

Figure 4: Color Transformation.

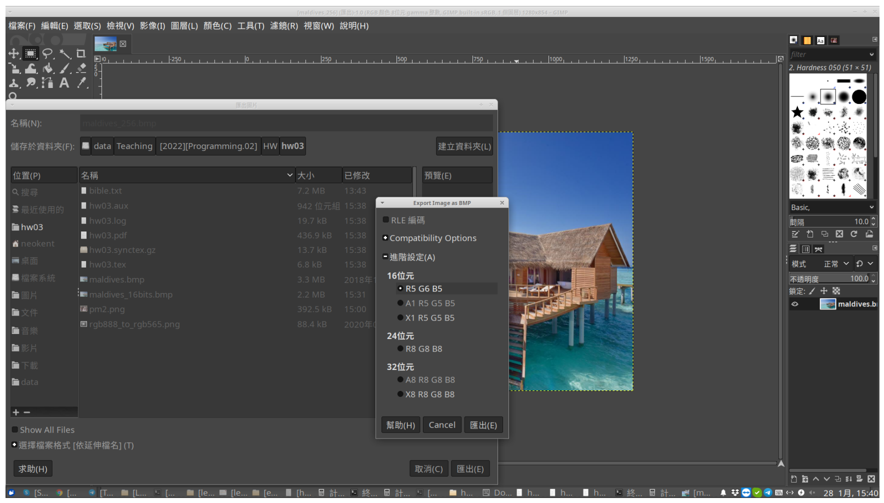

Figure 5: GIMP.

- The first line contains two positive integers m, n, which means the width is split into m parts and the height is split into n parts.
- The second line has one positive integers k, which means there will be k blocks below. Each block is the state of the sliding puzzle.
- Each block is has n rows where each row has m integers. The numbering start from 1 and is from the top left to right and top to down. Note that 0 means the removed piece.
- The first block is the initial state.
- The other blocks is **one move** from the above state.

The operation of this program is as follows.

```
$ ./hw0305
Please enter the image: doraemon.bmp
Please enter the puzzle: steps.txt
```

You need to output k BMP files where their names are 0.bmp, 1.bmp, . . ., k.bmp. Figure 6 is an example. Note that each piece should be surrounded with 3 pixels black line. If the generation setup text file is invalid, print an error message and terminate your program.

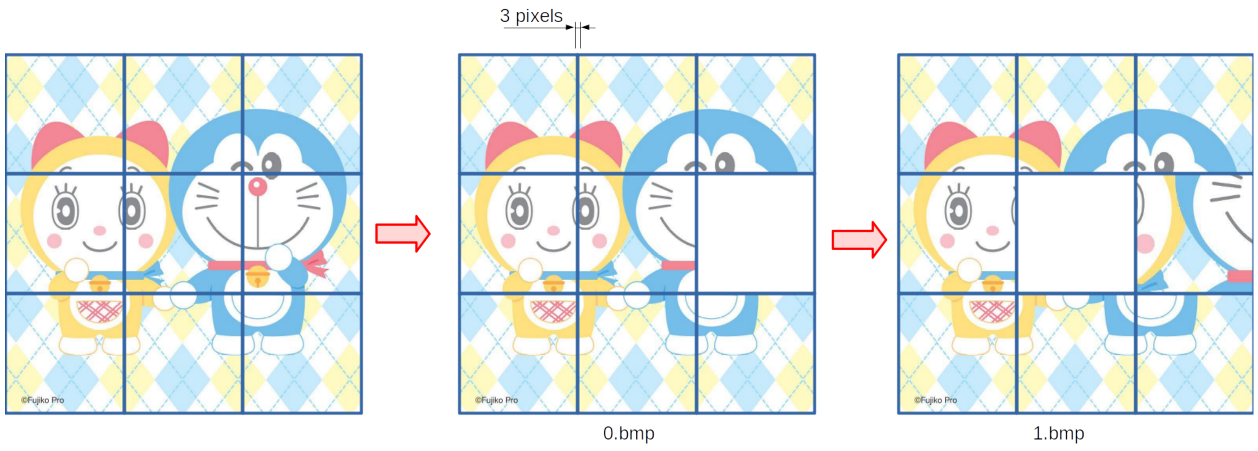

Figure 6: Sliding Puzzle.

6 Bonus: setjmp and longjmp (5 pts)

```
int setjmp(jmp_buf env);
void longjmp(jmp_buf env, int val);
```

What are these two functions used for? Please describe it and give an example code. Do not forget to look up what **jmp_buf** is.