C Programming II
2023 Spring
Homework 03

Instructor: Po-Wen Chi

Due: 2023.05.09 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.
- 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 Stream Cipher (20 pts)

Let's encrypt data. Given a file and a key, please develop a program to encrypt the file with the key. Do not worry, we just use the easiest way. A file F is denoted as a byte array $[f_0, f_1, \dots, f_{n-1}]$ where f_i is i-th byte of F. A key file K is denoted as a byte array $[k_0, k_1, \dots, k_{m-1}]$ where k_i is i-th byte of K. The ciphertext C is generated as follows:

$$\forall i \in \{0, \dots, n-1\}, c_i = f_i \oplus k_{i \bmod m}.$$

We use modular operation here because in general, the file size is much longer than the key size.

The program usage should be

```
$ ./hw0301 -e -k key_file -i input_file -o cipher
$ ./hw0301 -d -k key_file -i cipher -o output_file
```

2 ZIP (20 pts)

ZIP is an archive file format that supports lossless data compression. A ZIP file may contain one or more files or directories that may have been compressed. This time, I want you to develop a ZIP file utility. Do not worry, I will not ask you to decompress a ZIP file. What I want you to do is to display the directory structure in the ZIP file.

```
$ ./hw0302 test.zip
+-- test/ --+-- a/ --+-- hello
|           |        |
|           |        +-- world
|           |
|           +-- b/ --+-- MadokaBD.jpg
|                    |
|                    +-- anime.jpg
|
+-- game.png
```

You should support the following two options:

- **-a**: **default**, the display order should be file name ASCII ascending order.
- **-d**: the display order should be file name ASCII descending order.

Note that the **directories must be displayed before files.**

3 Mosaic (20 pts)

Given a BMP file, please develop a program to make a given area mosaic, as shown in Fig. 1.

The mosaic algorithm is simple. Given a number n, replace the block of $n \times n$ pixels with the average RGB value. Fig. 2 is a 2×2 block example. For your convenience, you can simply make those redundant pixels remain their original RGB values.

The program usage should be

```
$ ./hw0303 -x 500 -y 300 -w 100 -h 100 -n 10 -i spy.bmp -o spy2.bmp
$ ./hw0303 --help
```

The options are

- **-x**: **mandatory**, x value.
- **-y**: **mandatory**, y value.

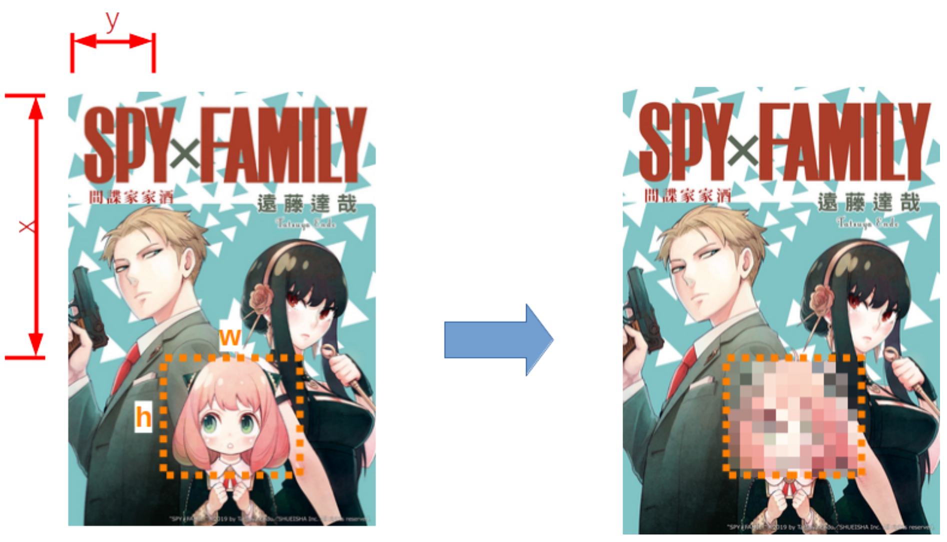


Figure 1: Example. The units of x, y, w, h are all pixels. Note that this is an example only and is not generated from the program of this assignment.

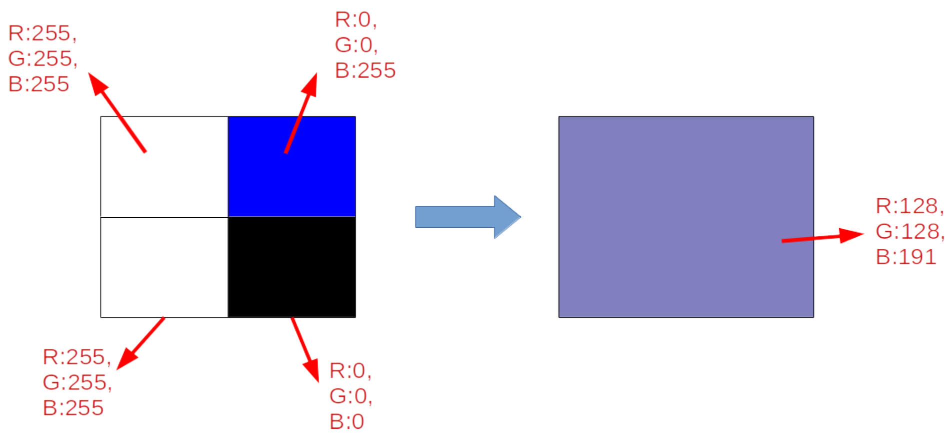
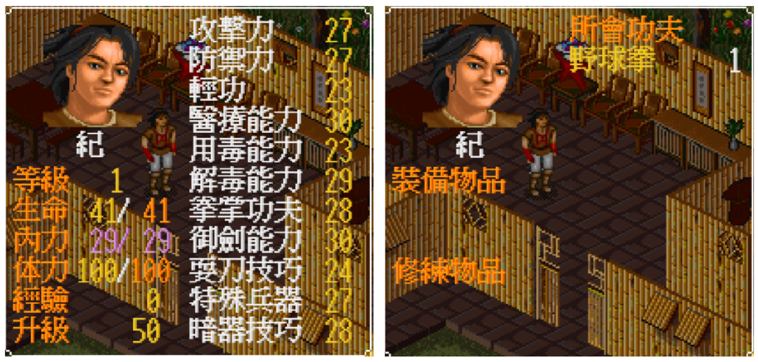


Figure 2: Mosaic example for a 2×2 block.

- **-w, −−width**: **mandatory**, w value.
- **-h, −−height**: **mandatory**, h value.
- **-n, −−number**: **optional**, n value. Default n is 2.
- **-i, −−input**: **mandatory**, the input BMP file name.
- **-o, −−output**: **optional**, the output BMP file name. Default name is **output.bmp**.
- **–help**: print the usage help.

For all invalid usage, you should give a warning message and terminate the program.



```
紀
等級      1   攻擊力   27
生命  41/ 41   防禦力   27
內力  29/ 29   輕功     23
体力 100/100   醫療能力 30
經驗      0   用毒能力 23
升級     50   解毒能力 29
              拳掌功夫 28
              御劍能力 30
              耍刀技巧 24
              特殊兵器 27
              暗器技巧 28
```

```
紀
所會功夫
野球拳    1
裝備物品
修練物品
```

Figure 3: The player's statistics.

4 Game Cheater: Heroes of Jin Yong (20 pts)

飛雪連天射白鹿
笑書神俠倚碧鴛

Jin Yong (Chinese: 金庸), was a Chinese wuxia (”martial arts and chivalry”) novelist and essayist. I believe that you all have read or heard his works, like **The Legend of the Condor Heroes (射鵰英雄傳)**, **04 The Return of the Condor Heroes (神雕俠侶)**, **Demi-Gods and Semi-Devils (天龍八部)** and so on. Heroes of Jin Yong (金庸群俠傳), first published in 1996, is a tactical role-playing PC game based on the storyline and characters in Jin Yong's Wuxia novels. In the game, the player takes on the role of present-day protagonist, who wakes up one day to find himself in the ancient Chinese Jianghu (martial arts world). The player learns that in order to return to modern society, he must find all the fourteen novels by Jin Yong and be declared champion of the Jianghu. The books are scattered around the Jianghu and the player must interact with many characters from the novels, each with his / her own story to tell.

You can download the game from the following URL and play the game in **dosbox**.

`https://archive.org/details/heros_of_jin_yong`

Unfortunately, as your teacher, I do not have enough time to enjoy this game. So I want to cheat!! Figure 3 is the player's abilities and the arts. Please develop a program to modify the player's **attributes**, **money**, **items** and **arts** stored in the save file. The interface is designed by yourself. The program name must be **hw0304**.

5 Cool Sound Infinity Exporter (20 pts)

Your TA *LoveSnowEx* loves singing, but he has been swamped with assignments lately, and hasn't had time to go to KTV to sing for a long time. You feel sorry for him, so you want to help him find his musical soul.

You have a good idea, which is to design a ***CSIE***, aka ***Cool Sound Infinity Exporter*(酷奬的無限聲音導出器)**, that can automatically generate accompaniment music files, so that *LoveSnowEx* can sing while working on assignments. You need to help him design ***CSIE*** so that it can read the musical notation text file that *LoveSnowEx* inputs and automatically generate suitable accompaniment music according to his music style and singing style.

```
$ ./hw0305
Welcome to Cool Sound Infinity Exporter!
Please enter input and output file name.
Input file name: easy.txt
Output file name: easy.mid
```

Here are some rules:

1. The input file consists of numbers and symbols, where numbers represent musical notes and symbols represent duration and pitch. Specifically, the numbers "`1`" to "`7`" represent *Middle Do* to *Middle Si*, "`0`" represents a rest, and "`-`", "`.`", "`;`", "`'`", and "`,`" represent changes in duration and pitch.

2. Specifically, the symbols "`-`", "`.`", and "`;`" denote double duration, one and a half duration, and half duration respectively. The symbols "`'`", and "`,`" denote an octave up and an octave down respectively. For instance, "`-`" represents a half note, and "`--`" represents a whole note; "`.`" represents a dotted note; "`;`" represents an eighth note, and "`;;`" represents a sixteenth note; "`'`" represents a note one octave higher, such as "`1'`", and "`,`" represents a note one octave lower, such as "`1,`".

3. The duration symbol will come after the octave symbol, e.g., "`1''--`".

4. Please ignore all white space characters in the input file, such as `' '` and `'\n'`.

5. For your simplicity, the output will be set to 120 BPM, which is the default MIDI tempo. There is no need for you to make any changes.

6. There may be many instruments, and each instrument will be marked as "`[INSTRUMENT]`" at the beginning of its respective part.

7. The input of the program is a "`.txt`" file of numerical notation, and the output is a "`.mid`" file.

8. I guarantee that there are no wrong inputs, so you don't need to worry about that.

You may not know what MIDI is, but don't worry. Here are some useful sites to help you solve this problem:

- Standard MIDI-File Format Spec. 1.1
- Standard MIDI File
- Online MIDI Player
- Musescore

I will provide you with example input and output files, named "`easy.txt`" and "`easy.mid`" respectively, and you can try "`hard.txt`" for testing multiple instruments. The song used in the example is "Twinkle, Twinkle, Little Star", which is one of my favorite songs!

6 Bonus: Wildcard (5 pts)

In this class, I have show you how to implement a process supporting lots of options and inputs. I believe that you all know how to do this. However, look the following scenario:

```
$ rm *.c
```

Suppose **rm** will receive only one argument ***.c** and there is no file called *.c. What happens here? Please describe how your computer works and design a lab to prove your description.