# Dash - MiniCV

## MiniCV

*Summary: bitmap, image processing*

# Contents

# Chapter I

# Introduction

Image processing is already used in many fields. Medical field, Transmission and encoding, Robot vision, Pattern recognition etc. Did you know that you can handle bitmap images in C without using OpenCV? This project offers you an opportunity to learn about bitmap structure and image processing basics. We will provide the bitmap image you will use for the project.

The beautiful palace in the picture is Gyeongbokgung Palace in South Korea. Gyeongbokgung Palace was built in 1395, burned down by the war in 1592, and re-built in 1868. Although most of the buildings in the palace have disappeared, the main buildings remain, and it is an important historical site where you can check the appearance of the royal palace in South Korea. However, the image of this beautiful Gyeongbokgung Palace is ruined. You need to restore this image.

# Chapter II

# General rules

- Your project must be written in according with the Norm. If you have bonus files/ functions, they are included in the norm check and you will receive a 0 if there is a norm error inside.

- Your functions should not quit unexpectedly (segmentation fault, bus error, double. free, etc) apart from undefined behaviors. If this happens, your project will be considered non functional and will receive a 0 during the evaluation.

- All heap allocated memory space must be properly freed when necessary. **No leaks will be tolerated.**

- If the subject requires it, you must submit a Makefile which will compile your source files to the required output with the flags -Wall -Wextra and -Werror, and your Makefile must not relink.

- Your Makefile must at least contain the rules $(NAME), all, clean, fclean, and re.

- We encourage you to create test programs for your project even though this work won't have to be submitted and won't be graded. It will give you a chance to easily test your work and your peers' work. You will find those tests especially useful during your defence. Indeed, during defence, you are free to use your tests and/or the tests ofthe peer you are evaluating.

# Chapter III

# Mandatory part

## Exercise 00: RGB to BGR

|  | Exercise : 00 |
|---|---|
| Program name : RGB2BGR | |
| Turn-in directory : *ex*00/ | |
| File to turn in : Makefile, *.c */*.c, *.h, */*.h | |
| Allowed functions : fopen, fread, fwrite, fclose, fprintf, fseek, malloc, free, exit | |
| Description : Restore the given bmp file to the original image | |

The color of the picture has changed.

Write a program that restores the original picture.

Your program must save the **original.bmp** file.

The evaluation is conducted by comparing the original bitmap file with the recovered bitmap file.

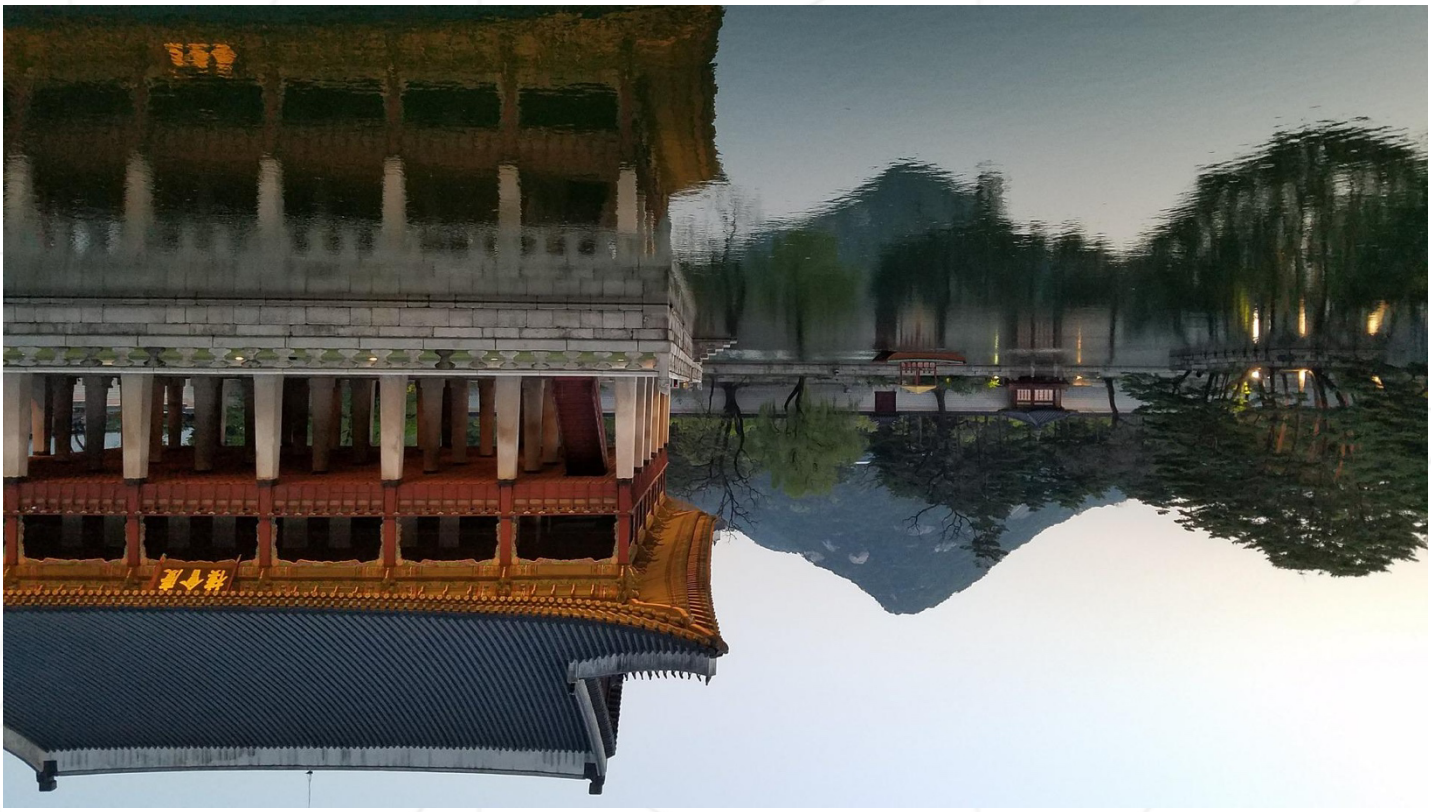
Apply the same rule to all subsequent exercises.

You have to use your own bitmap structure.

bitmap file header / bitmap info header
Do you now the -fpack-struct compiler flag?

# Exercise 01: Upside down / Right to left

|  | Exercise : 01 |
|---|---|
| Program name : reverse | |
| Turn-in directory : *ex*01/ | |
| File to turn in : Makefile, *.c */*.c, *.h, */*.h | |
| Allowed functions : fopen, fread, fwrite, fclose, fprintf, fseek, malloc, free, exit | |
| Description : Restore the given bmp file to the original image | |



# Chapter IV

# Bonus part

## Exercise 02: Zoom in

|  | Exercise : 02 |
|---|---|
| Program name : zoom | |
| Turn-in directory : *ex*02/ | |
| File to turn in : Makefile, *.c */*.c, *.h, */*.h | |
| Allowed functions : fopen, fread, fwrite, fclose, fprintf, fseek, malloc, free, exit | |
| Description : Write a program that magnifies a original bitmat file | |

**You can pass this dash-project without doing exercise 02.**

Don't spend too much time on bonus.

There are many ways to zoom in, but the choice is freedom.

Write a program that stores a given bitmap file to double magnification.