THE UNIVERSITY OF SYDNEY

INFO1910 Assignment 1

Due: 11:59PM Sunday 2 May 2021 local Sydney time

This assignment is worth 20% of your final assessment

Background

The Peak Signal-to-Noise Ratio is a measurement used in relation to the corruption of a signal or image by some noise. We can consider an image as an array of pixels, each pixel being represented by an integer between 0 and 255 (inclusive). If I is an array of pixel values of length n representing an image, and C is an array representing a corrupted version of that image, then we can define the *mean squared error* between the images as follows:

$$MSE = \frac{1}{n}\left(\sum_{i=1}^{n}(I[i] - C[i])^2\right)$$

Once we have the MSE, we can define the Peak Signal-to-Noise ratio, or PSNR:

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right)$$

This definition makes sense for an array containing one-dimensional pixels (each pixel has only a single numeric value), meaning that the above images are monochrome. Pixels in colour images are made up of three different values each. In order, they represent the pixel's red, green, and blue components. If I and C are two-dimensional arrays representing coloured images (the outer array containing the pixels in order, and each of the inner arrays being a single pixel of length 3), then we can instead define the MSE as

$$MSE = \frac{1}{3n}\left(\sum_{j=1}^{3}\sum_{i=1}^{n}(I[i][j] - C[i][j])^2\right)$$

Then the same definition of PSNR holds.

Notice that if the two images for which PSNR will be calculated are identical, then MSE will be 0 and the given definition of PSNR is undefined due to zero division. In this case we define PSNR as 0.

Task Description

Your assignment will be to write an interactive menu in Python capable of loading data which represents images, and calling functions in both Python and C which will compute variants of the PSNR between pairs of those images. You are provided with the following files:

`menu.py`

This will be the entry point of your program, and will contain the code you write to manage the interactive menu.

`py_functions.py`

You will fill out the python functions included in this file to compute different PSNR variants.

`c_functions.c`

You will fill out the C functions included in this file to compute different PSNR variants.

`c_functions.h`

This header file has been provided so that all functions and structs for your C code are already declared.

`translation.py`

This file exists to facilitate calling C functions from a python program. **You will not need to modify this file at all.**

Note that in example code in this document, user input will be indicated by '>'. Your program should not print '>' at any time.

The Menu

Upon startup, the menu will print

```
Welcome to the PSNR image menu!
```

It will then proceed to print

```
--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands
```

before waiting for the user to give a command. Every time a command completes, whether successfully or not, the second of these messages will print again before the program accepts a new command. Note that if the user switches mode between Python and C, this message will need to update so as to correctly display the current mode.

Commands are provided by the user typing the relevant word into the terminal and pressing 'enter'. Commands are case-insensitive and ignore whitespace, but must contain exactly the correct characters. For instance,

```
>help

>        HELP
```

and

```
>hElP
```

would all be successful in entering the 'help' command, but

```
>help me please
```

would not.
If the given command is not recognised, the menu will print

```
Invalid command.
```

and wait for a new command to enter. If no valid command is received, the menu header message will not be reprinted.

When the `help` command is received, the menu will print

```
Commands:
load: Load a single image into the program for use
show: Display all images currently loaded
psnr-r: Calculate the PSNR between the red values for two colour images
psnr-g: Calculate the PSNR between the green values for two colour images
psnr-b: Calculate the PSNR between the blue values for two colour images
psnr: Calculate the PSNR between all values for two images
mode: Toggle mode between C and Python
help: Print out this command list
quit: Exit the PSNR Image Menu
```

The `load` command reads an image out of a file and stores it in program memory so that it can be used. See the **Loading** section for more details.

The `show` command displays a summary of all images which have been loaded so far. For example:

```
Mode: Python
Type 'help' to see all commands

>show
Loaded images:
Image 1, Length 3, colour.
Image 2, Length 5, monochrome.
Image 3, Length 3, colour.
Image 4, Length 5, monochrome.
Image 5, Length 10, colour.
Image 6, Length 14, monochrome.
Image 7, Length 10, colour.
Image 8, Length 14, monochrome.
Image 9, Length 10, colour.
Image 10, Length 14, monochrome.
Image 11, Length 10, colour.
Image 12, Length 14, monochrome.
```

Note that image length (number of pixels) and type is displayed, but no facility exists in the menu for giving the images custom names; their index numbers are based solely on the order in which they were loaded.

The commands `psnr-r`, `psnr-g`, and `psnr-b` each compute the PSNR of the red, green, and blue components respectively between two images. See the **Calling PSNR Commands** section for more details. Note that because these three compute PSNR for a specific colour only, they cannot accept monochrome images.

The `psnr` command computes PSNR between two images. It can accept either monochrome or colour images, but cannot compute PSNR between one colour and one monochrome image. It computes the full PSNR of colour images, taking account of all three colour components.

The `mode` command toggles between C and Python modes. The mode the menu is in determines whether it will use C or Python functions when any PSNR calculation is requested. `mode` requires no arguments and does not ask for additional information once invoked; it immediately toggles. For example:

```
--- PSNR Image Menu ---

Mode: Python
```

```
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands
```

The `quit` command immediately exits the program. Nothing else is printed.

Loading

Images can be provided to the program in the form of files. Each line in an image file represents one pixel. For a monochrome image, each line will have only a single integer number on it. For a colour image, each line will have three integer numbers separated by commas, representing that pixel's red, green, and blue values respectively. For example, a 3-pixel long colour image file will look like this

```
1,1,1
2,3,4
7,8,9
```

and a 5-pixel monochrome image file will look like this

```
1
8
5
9
2
```

No other file format is valid for the PSNR image menu to read. All entries in the file must be integers, and must be within the range of 0 to 255 inclusive.

Upon invoking the `load` command, the user will be asked to enter the name of the file to load:

```
Enter the filename you want to load:
```

If the given file does not exist, the program will print

```
Error: File does not exist.
```

If the file exists but has no contents (there are no lines to read), the program will print

```
Error: File is empty.
```

If the file is not entirely made up of three-part comma-separated lines, or if the file is not entirely made up of lines containing one entry only, the program will print

```
Error: Image does not appear in RGB or monochrome format.
```

If something which cannot be interpreted as an integer appears in one of the fields, the program will print

```
Error: Non-integer value found in image file.
```

If there exists in the file an integer value which is outside the range of 0 to 255, the program will print

```
Error: Number outside the range of 0 to 255 found in image file.
```

If any of these error conditions is met, the loading will fail and the program will return to the main menu. If more than one of these error conditions exists in the file, the program will present the error which would be found first if reading through the file left to right, then top to bottom.

If none of these error conditions exists in the file, the image will be successfully loaded into program memory. It will now appear in the list printed by the `show` command, and can be given to any `psnr` command calculations. You are free to represent files in program memory however you want, but be aware you will need them in a specific format to call the C versions of the PSNR functions.

Calling PSNR Commands

Upon calling any of the four PSNR commands from the menu, the program will ask

```
What is the index of the image you would like to select?
```

If however no images have yet been loaded, then the program will not ask for an index at all, instead printing

```
No images have been loaded. No image can be selected.
```

Once an index has been requested, if the user inputs something that is not an integer, the program will output

```
That is not a valid integer.
```

and ask for the index again. If the given integer is not between 1 and the number of images that have been loaded, the program will also inform the user and ask for an index again. For instance, if 4 images have been loaded, the program would print

```
The index should be between 1 and 4.
```

Notice the indexing starts at 1, not at 0.

The program will continue to ask for an index until it has received two valid indexes, for the two images that will be compared. Once two valid indexes have been received, the command will proceed. If the requested PSNR function was any of the single-colour commands, and either or both of the given indexes is for a monochrome image, the program will print one of the following

```
One of those images is not in colour; cannot compute red PSNR.
One of those images is not in colour; cannot compute green PSNR.
One of those images is not in colour; cannot compute blue PSNR.
```

If `psnr` was the command given, and one of the given images is colour and one is monochrome, then the program will print

```
Images are not the same type; cannot compute PSNR between them.
```

If there are no problems with image type, but the two given images are not the same length, the program will print

```
Images are not the same length; cannot compute PSNR between them.
```

If none of the above errors are found, then the program will proceed to call the relevant python or C function to calculate the desired PSNR, based on which mode the menu is currently in. All of the error checking already specified should occur inside `menu.py`, not in the PSNR calculation functions.

Python PSNR functions

The file `py_functions.py` contains all the functions you will write to compute the different kinds of PSNR. Do not write these functions in `menu.py`, leave them in `py_functions.py`. Once you have imported `py_functions` in `menu.py`, you will be able to call the PSNR functions as follows

```
red_psnr = py_functions.py_r_psnr(image1, image2)
green_psnr = py_functions.py_g_psnr(image1, image2)
blue_psnr = py_functions.py_b_psnr(image1, image2)
total_psnr = py_functions.py_total_psnr(image1, image2)
```

You must fill out these four functions in `py_functions.py` to correctly calculate the PSNR between the two images given. The functions you must fill out expect two arguments; one for each of the images. You may use any representation of these images you want.

C PSNR functions

The file `c_functions.c` contains C source code for the four functions you will write to compute PSNR variants in C. The functions you will need to write are

```
float c_r_psnr(struct Image* image1, struct Image* image2)
float c_g_psnr(struct Image* image1, struct Image* image2)
float c_b_psnr(struct Image* image1, struct Image* image2)
float c_total_psnr(struct Image* image1, struct Image* image2)
```

Notice there exist several other functions in the `c_functions.c` file. You will not need to modify or study these; they exist to build the structs your functions will work with. These functions, which you will not need to modify, are

```
void make_coloured_image(int* red, int* green, int* blue, int len,
        struct ColouredPixels* cp, union PixelData* pd, struct Image* im)
void make_monochrome_image(int* grey, int len, union PixelData* pd,
        struct Image* im)
float c_r_psnr_wrapper(int* red1, int* green1, int* blue1, int* red2,
        int* green2, int* blue2, int len)
float c_g_psnr_wrapper(int* red1, int* green1, int* blue1, int* red2,
        int* green2, int* blue2, int len)
float c_b_psnr_wrapper(int* red1, int* green1, int* blue1, int* red2,
        int* green2, int* blue2, int len)
float c_total_psnr_wrapper(int* red1, int* green1, int* blue1,
        int* red2, int* green2, int* blue2, int len, int is_coloured)
```

The four functions you will need to write accept two structs, each representing one of the images between which PSNR will be calculated. These structs are defined in `c_functions.h`. Notice the structs contain unions, which means they can represent either colour images or monochrome images. The field `is_coloured` inside the `Image` struct is a boolean value specifying which type the given image is.

To call your C functions from inside `menu.py`, you will need to use the `translation.py` file provided. Once you have imported `translation` in `menu.py`, you can call the following functions:

```
red_psnr = translation.call_c_r_psnr(image1, image2)
green_psnr = translation.call_c_g_psnr(image1, image2)
blue_psnr = translation.call_c_b_psnr(image1, image2)
total_psnr = translation.call_c_total_psnr(image1, image2)
```

Each of these functions will take the image objects given to them and call the C wrapper functions in `c_functions.c`. These wrapper functions build the structs for your PSNR functions, and then call the PSNR functions, providing the finished structs to them. They also transmit the return value from your C functions back to the original translation function calls. In effect, when you call

```
total_psnr = translation.call_c_total_psnr(image1, image2)
```

the variable `total_psnr` will then contain whatever float value your function `c_total_psnr` returned.

The 'call' functions in `translation.py` expect a 2 dimensional list (a list of lists) to represent each image. Each of the inner lists is one pixel; if the image is colour, these lists will be length 3, and if the image is monochrome, they will have length 1. All the entries in all the pixels should be integers. The outer list contains all the pixels in order; the length of the outer list equals the number of pixels in the image. The 'call' functions should also accept 2 dimensional tuples instead of lists, but make sure to test it if you want to do this.

Compilation and Execution

You will need to provide a makefile along with your submission. In order for `translation.py` to access your C functions, you will need to perform a compilation something like this:

```
gcc -fPIC -shared -Wall -Werror -o c_functions.so c_functions.c
```

It is crucial that the final product of all your C code is named `c_functions.so`; this is the file name `translation.py` is looking for. Make sure your code compiles correctly on Ed; if your C code cannot compile, you cannot receive marks for it.

The program will always be executed as follows:

```
python3 menu.py
```

No command line arguments will be used. This means you can use command line arguments for debugging purposes if you so desire.

Hints and Restrictions

Notice that `py_functions.py` imports `math` and `c_functions.h` includes `math.h`. These give you access to the `math.log10` function in Python, and the `log10` function in C. These are the functions you are expected to use for calculating logarithms. `math.h` also gives you access to the `pow` function for calculating exponentials.

You may not use any modules which trivialise any part of the assignment. For instance, the `pandas` module includes functions to read comma-separated files automatically, and the `numpy` module makes it easy to perform an operation to every cell of an array with only one instruction. Tutors have final say in what counts as 'trivialising' a task. A good rule of thumb is that if it is possible for

you to write your own code to do a job using the existing course content, you should write your own code rather than use a module. If you use modules to do tasks for you, you may receive no marks for part or all of your assignment's correctness.

Examples

Most of the errors that can be encountered when loading a file

```
Welcome to the PSNR image menu!

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >non_existent_file
Error: File does not exist.

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >empty_image
Error: File is empty.

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >float_pixel1
Error: Non-integer value found in image file.

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >invalid_format1
Error: Image does not appear in RGB or monochrome format.
```

```
--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >non_int_pixel1
Error: Non-integer value found in image file.

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >out_of_range_pixel1
Error: Number outside the range of 0 to 255 found in image file.

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

quit
```

Some errors that can arise when trying to select a file for PSNR computation

```
Welcome to the PSNR image menu!

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>show
No loaded images to show.

--- PSNR Image Menu ---
```

```
Mode: C
Type 'help' to see all commands

>psnr
No images have been loaded. No image can be selected.

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_colour1

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_colour1_corrupt

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_monochrome1

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_monochrome1_corrupt

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>show
Loaded images:
Image 1, Length 3, colour.
Image 2, Length 3, colour.
```

```
Image 3, Length 5, monochrome.
Image 4, Length 5, monochrome.

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>psnr-r
What is the index of the image you would like to select? >100
The index should be between 1 and 4.
What is the index of the image you would like to select? >3
What is the index of the image you would like to select? >An index
That is not a valid integer.
What is the index of the image you would like to select? >4
One of those images is not in colour; cannot compute red PSNR.

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>psnr-g
What is the index of the image you would like to select? >1
What is the index of the image you would like to select? >-1
The index should be between 1 and 4.
What is the index of the image you would like to select? >3
One of those images is not in colour; cannot compute green PSNR.

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>psnr
What is the index of the image you would like to select? >5
The index should be between 1 and 4.
What is the index of the image you would like to select? >2
What is the index of the image you would like to select? >0
The index should be between 1 and 4.
What is the index of the image you would like to select? >4
Images are not the same type; cannot compute PSNR between them.

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands
```

```
>quit
```

Examples of several PSNR calculations

```
Welcome to the PSNR image menu!

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_colour1

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_colour1_corrupt

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_colour1_corrupt

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>load
Enter the filename you want to load: >small_monochrome1_corrupt

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>show
Loaded images:
Image 1, Length 3, colour.
```

```
Image 2, Length 3, colour.
Image 3, Length 5, monochrome.
Image 4, Length 5, monochrome.

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>psnr-r
What is the index of the image you would like to select? >1
What is the index of the image you would like to select? >2
Red PSNR: 40.34929110484267

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>psnr-g
What is the index of the image you would like to select? >2
What is the index of the image you would like to select? >1
Green PSNR: 46.88141632080078

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>psnr-b
What is the index of the image you would like to select? >1
What is the index of the image you would like to select? >1
Blue PSNR: 0.0
```

```
--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>psnr
What is the index of the image you would like to select? >3
What is the index of the image you would like to select? >4
PSNR of images: 45.57807922363281

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>psnr
What is the index of the image you would like to select? >4
What is the index of the image you would like to select? >3
PSNR of images: 45.578078557646045

--- PSNR Image Menu ---

Mode: Python
Type 'help' to see all commands

>mode

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands
```

```
>psnr
What is the index of the image you would like to select? >2
What is the index of the image you would like to select? >2
PSNR of images: 0.0

--- PSNR Image Menu ---

Mode: C
Type 'help' to see all commands

>quit
```

Marking Criteria

The following is the marking breakdown, each point contributes a portion to the total 20% of the assignment.

Marks are allocated on the basis of:

- Correctness - 14/20
- Code style - 6/20

Correctness

- Public testcases - 8/14
- Hidden testcases after due date - 6/14

There will be some leniency for testcases which print long float values.

Code style

- Code structure: breaking tasks up into functions - 2/6
- Code clarity: variable and function names - 2/6
- Comments: explain complex operations, but do not use too many - 2/6

Warning: Any attempts to deceive or disrupt the marking system will result in an immediate zero for the entire assignment. Negative marks can be assigned if you do not follow the assignment description or if your code is unnecessarily or deliberately obfuscated.

Academic declaration

By submitting this assignment you declare the following:

I declare that I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure, and except where specifically acknowledged, the work contained in this assignment/project is my own work, and has not been copied from other sources or been previously submitted for award or assessment.

I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to severe penalties as outlined under Chapter 8 of the University of Sydney By-Law 1999 (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgment from other sources, including published works, the Internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

I realise that I may be asked to identify those portions of the work contributed by me and required to demonstrate my knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

I acknowledge that the School of Computer Science, in assessing this assignment, may reproduce it entirely, may provide a copy to another member of faculty, and/or communicate a copy of this assignment to a plagiarism checking service or in-house computer program, and that a copy of the assignment may be maintained by the service or the School of Computer Science for the purpose of future plagiarism checking.