

Special Effects

COMP0015 Term 2 Coursework 1 – 25% of the module

This document explains the arrangements for the coursework. You will work in pairs to create an application for modifying images in PPM format according to the specification set out in this document.

Deadline

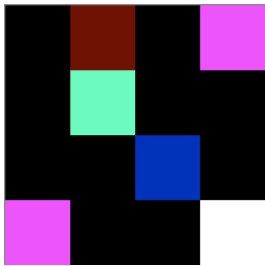
Midday Monday 2nd March 2020.

Overview

Most people are familiar with the fact that a digital image is a grid comprised of thousands of pixels with each pixel containing a single colour. The number of pixels in an image determines the definition of the image. You have been learning Python on this course for 3 weeks and you know enough to be able to process an image pixel by pixel and transform it in some way to produce some simple effects.

This may seem daunting but we will be using a very simple image format called PPM (or Portable Pix Map). PPM images are not really used any more but they're useful to us because a PPM image is encoded as human-readable ASCII text. For those of you who want to know more, the formal image specification can be found [here](#). It's not necessary to read it this for the coursework.

Here is a sample ppm file and the corresponding image. It has two parts:

<pre>P3 4 4 255</pre>	<p>This part of the image is the header. The second row is important, it contains the number of rows and columns in the image.</p>
<pre>0 0 0 100 0 0 0 0 0 255 0 255 0 0 0 0 255 175 0 0 0 0 0 0 0 0 0 0 0 0 0 15 175 0 0 0 255 0 255 0 0 0 0 0 0 255 255 255</pre> 	<p>This part represents the pixels. This image has 4 x 4 (16) pixels. Each pixel has a red, green and blue (r, g, b) value.</p> <p>0, 0, 0 means no colour (black) and 255, 255, 255 is the maximum level of a colour (white). Take a look at the RGB calculator to experiment with RGB values here.</p>

Keep in mind, each square is one pixel, so the real thing is much smaller (this image was blown up by 5000%).

Viewing PPM files

You will need some software on your computer to:

- View a plain text file
- View an image in PPM format

It's likely you have some choices here. Here's some initial suggestions:

Software	View plain text	View image	Platforms
Any text editor such as Notepad, atom.	✓		All
IDLE	✓		Windows, MacOS UCL Desktop
PyCharm https://www.jetbrains.com/pycharm-edu/download/index.html#section=windows		✓	Windows, MacOS
IrfanView https://www.irfanview.com/		✓	Windows, UCL Desktop
Gimp https://www.gimp.org/		✓	Windows, MacOS UCL Desktop
Website: https://www.kylepaulsen.com/stuff/NetpbmViewer/		✓	

Your Assignment

You are provided with some starter code and some sample pictures which you must download and save before starting the assignment. The pictures are relatively small in size which is why they have been chosen.

Functions

The skeleton code contains these empty functions. You must complete them.

Function names	Description
<code>greyscale</code>	In this function you will work on a pixel at a time. Take the average of the red, green and blue value in a pixel and replace the red, green and blue values with the average.
<code>only_red</code> , <code>only_blue</code> , <code>only_green</code>	These functions remove two of the colour channels in the image. A colour channel is: all the red values in an image or all the green values or all the blue values. <code>only_red</code> removes the green and blue colour channel.
<code>negative_red</code> , <code>negative_blue</code> , <code>negative_green</code>	These three functions do the same thing to each of the colour channels. A colour channel is: all the red values in an image or all the green values or all the blue values. To make a channel negative, you subtract its current value from 255.
<code>negative_image</code>	You can create a negative image by replacing the red, green and blue values with their negative value. Use the functions above to do this.
<code>extreme_contrast</code>	In this function you will alter every pixel so that: if it is less than 127, you set it to zero and if it is greater than or equal to 127, you set it to 255.

Testing:

Your functions should NOT cause an r, g, b value to be less than 0 nor larger than 255. All pixel values must be integers.

Submitting your assignment

At the submission link on moodle:

1. Make sure your student numbers (not your names) are included in comments at the top of your program.
2. Upload your program.
3. If you have included any additional functionality not specified in the brief, please submit a short document describing what you have done. Keep this short, it is not graded. Upload all additional materials.

Assessment

You are expected to show that you can code competently using the programming concepts covered so far in the course including (but not limited to): use of variables, conditions, loops, functions and lists.

Marking criteria will include:

- Correctness – your code should perform as specified
- Programming style – your variable names should be meaningful and your code as simple and clear as possible. See section Style Guide further on for more detail.
- Your assignment will be marked using the rubric at the end of this document. This is the standard rubric used in the Department of Computer Science. Both members of the team will receive the same mark unless the lecturer has been notified of issues and determines otherwise. Marks for your project work will be awarded for the capabilities (i.e. functional requirements) your system achieves, and the quality of the code.

Additional Challenges

- Additional marks may also be gained by taking on extra challenges but you should only attempt an additional challenge if you have satisfied all requirements for the coursework.
- It's up to you what you choose to do, if anything. You could add additional special effects using more menu options.
- Likewise, you could write some automated tests for your functions using `pytest` or, more simply by using `assert` statements.

Plagiarism

Plagiarism will not be tolerated. Your code will be checked using a plagiarism detection tool.

Style Guide

You must adhere to the style guidelines in this section.

Formatting Style

1. Use Python style conventions for your function and variable names (pothole case: lowercase letters with words separated by underscores (`_`) to improve readability).
2. Choose good names for your functions and variables. For example, `num_bright_spots` is more helpful and readable than `nbs`.
3. Use a tab width of 4 or 8. The best way to make sure your program will be formatted correctly is never to mix spaces and tabs -- use only tabs, or only spaces.
4. Put a blank space before and after every operator. For example, the first line below is good but the second line is not:


```
b = 3 > x and 4 - 5 < 32
```

```
b= 3>x and 4-5<32
```
5. If you add functions, write a docstring comment for each function. (See below for guidelines on the content of your docstrings.) Put a blank line after every docstring comment.
6. Each line must be less than **80 characters** long *including tabs and spaces*. You should break up long lines using `\`.

Docstrings

If you add your own functions you should comment them using docstrings. Take a look at the code you've been given for some examples. Your comments should:

1. Describe precisely *what* the function does.
2. Do not reveal *how* the function does it.
3. Make the purpose of every parameter clear.
4. Refer to every parameter by name.
5. Be clear about whether the function returns a value, and if so, what.
6. Explain any conditions that the function assumes are true. Examples: "n is an int", "n != 0", "the height and width of p are both even."
7. Be concise and grammatically correct.
8. Write the docstring as a command (e.g., "Return the first ...") rather than a statement (e.g., "Returns the first ...")

UCL Computer Science: Marking Criteria and Grade Descriptors



	Fail		Pass (2:2)		Merit (2:1)		Distinction (1 st)	
	Inadequate	Weak	Satisfactory		Good		Excellent	Exceptional
	Below 40: BSc: Fail MEng: Fail	40-49: BSc: 3rd MEng: Fail	50-54: Low pass	55-59: High pass	60-64: Low merit	65-69: High merit	70-79	80-89 90+
1 Quality of the response to the task set: answer, structure and conclusions	Either no argument or argument presented is inappropriate and irrelevant. Conclusions absent or irrelevant.	An indirect response to the task set, towards a relevant argument and conclusions.	A reasonable response with a limited sense of argument and partial conclusions.		A sound response with a reasonable argument and straightforward conclusions.		A distinctive response that develops a clear argument and sensible conclusions, with evidence of nuance.	Exceptional response with a convincing, sophisticated argument with precise conclusions.
2 Understanding of relevant issues	Misunderstanding of the issues under discussion.	Rudimentary, intermittent grasp of issues with confusions.	Reasonable grasp of the issues and their broader implications.		Sound understanding of issues, with insights into broader implications.		Thorough grasp of issues; some sophisticated insights.	Exceptional grasp of complexities and significance of issues.
3 Engagement with related work, literature and earlier solutions	Very limited or irrelevant reading.	Significant omissions in reading with weak understanding of literature consulted.	Evidence of relevant reading and some understanding of literature consulted.		Evidence of plentiful relevant reading and sound understanding of literature consulted.		Extensive reading and thorough understanding of literature consulted. Excellent critical analysis of literature.	Expert-level review and innovative synthesis (to a standard of academic publications).
4 Analysis: reflection, discussion, limitations	Erroneous analysis. Misunderstanding of the basic core of the taught materials. No conceptual material.	Analysis relying on the partial reproduction of ideas from taught materials. Some concepts absent or wrongly used.	Reasonable reproduction of ideas from taught materials. Rudimentary definition and use of concepts.		Evidence of student's own analysis. Concepts defined and used systematically/effectively.		Evidence of innovative analysis. Concepts deftly defined and used with some sense of theoretical context.	Exceptional thought and awareness of relevant issues. Sophisticated sense of conceptual framework in context.
5 Algorithms and/or technical solution	No solution to the given problem, completely incorrect code for the given task.	Rudimentary algorithmic/technical solution, but mostly incomplete.	Reasonable solution, using basic required concepts, several flaws in implementation.		Good solution, skilled use of concepts, mostly correct and only minor faults.		Excellent algorithmic solution, novel and creative approach.	Exceptional solution and advanced algorithm/technical design.
6 Testing of solution (e.g., correctness, performance, evaluation)	No testing or evaluation done.	Few test cases and/or evaluation, but weak execution.	Basic testing done, but important test cases or parts of evaluation missing or incomplete.		Solid testing or evaluation of solution, well done evaluation with good summary of findings.		Very well done test cases, excellent evaluation and very high quality summary of findings.	Exceptionally comprehensive testing, extremely thorough approach to testing and/or evaluation.
7 Oral presentation or demonstration of solution	Poorly done presentation or demonstration, very low quality.	Ineffective oral presentation or demo of the solution.	Able to communicate, present and/or demonstrate solution and summarise work in appropriate format.		Overall good presentation or demo, persuasive and compelling.		Very high quality of delivery. Use of presentation medium with professional style.	Flawless and polished presentation, exceptional quality of demonstration.
8 Writing, communication and documentation	Style and word choice seriously interfere with comprehension.	Style and word choice seriously detract from conveying of ideas.	Style and word choice sometimes detract from conveying of ideas.		Style and word choice work well to convey most important ideas. Well documented.		Style and word choice show fluency with ideas and excellent communication skills.	Reads as if professionally copy edited. Exceptional high quality of writing.
9 Formatting aspects, visuals, references	Poorly formatted, inappropriate visuals, and incorrect reference formatting.	Formatting, visuals and referencing seriously detract from argument.	Formatting, visuals and referencing sometimes distract from argument.		Formatting well-done and consistent, good visuals and consistent referencing.		Formatting, visuals and referencing are impeccable.	Exceptional presentation, impeccable formatting of the document and references.

1-19: Misunderstanding of assignment or similar
20-29: 5 inadequate
30-39: 4 inadequate
34-39: 3 inadequate