ECOR 1042 PROJECT DESCRIPTION

Learning Outcomes

By the end of the course, students should be able to work in a small team to iteratively and incrementally design, implement and test a small-scale, interactive program that is partitioned into multiple modules, given a detailed specification of the functional requirements.

Overview

Students will work in pre-assigned teams of 4 (minimally 3). In most cases, one of the team members will be a student within the Department of Systems and Computer Engineering. This person is the team leader. As team leader, there is no expectation nor benefit from having the most experience in computer programming; instead, the team leader will have to exercise and demonstrate leadership in organization and delegation of duties, time-management, facilitation of meetings, potentially conflict management, and liaison with the teaching assistants and instructors.

The project itself is to build a **photo-editing program** that, among other features, allows a user to apply a variety of image *filters* to their images. Think of the camera on your phone: Have you ever played with the filters (or, photo effects) that turn your photos into black-and-white (monochrome), or bump up the contrast (to make the colours more vivid)? If not, check out your camera now, or click on the three links below to understand what your program will ultimately do:

- On Android phones
- On iPhones
- On Windows

To develop this project you will need to learn about *digital colour representation*, and the manipulation of pixels within images. Manipulating pixels is a large exercise in Python that involves tuples, conditional and iterative statements.

The code itself will be constructed as a collection of modules, with different team members writing different portions of the code. Modular code must be written according to exacting standards so that they fit together. Testing will be built into the code modules from the beginning. Through guided *milestones* (steps), the code will be written *incrementally* (slowly, bit-by-bit). At each milestone, something new will be added, forcing *iterative* re-writing of your existing code.

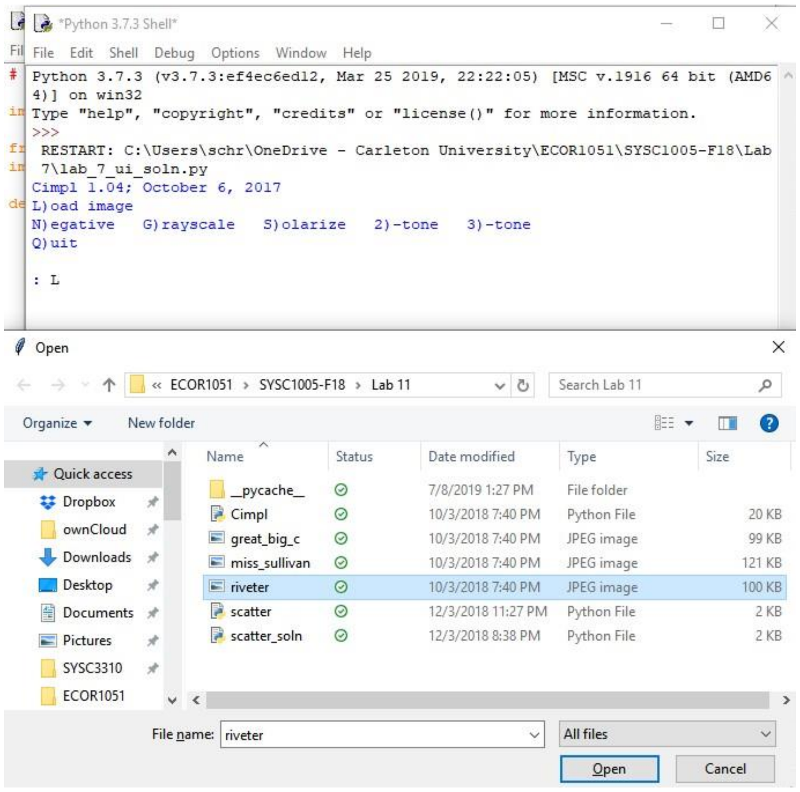
Time wise, the project is organized into **three milestones**. In engineering, a *milestone* is marked by something that can be tested and demonstrated. Each milestone is accomplished at the end of each two-week period and they will be posted in cuLearn.

Each milestone will have two elements:

- A Milestone Description:
  - A helpful, guided breakdown of the tasks to be completed, along with details of requirements for the documentation, testing, coding of the program modules.
  - Submission deadlines with the required set of deliverables. Some submissions will be done as individuals; some will be done as a team.
- A Milestone Rubric:
  - A clear outline of the marking expectations for the milestones.

Functional Requirements

In the photo-editing program, you will be able to load an image and then perform a sequence of cumulative filters and other editing on this image. The user interface will be text-based – prompting the user to enter one command after another. As a quick introduction (with much more information and detailed requirements coming later), consider the screenshot shown in Figure 1. Users will load a file (in this case, using the L command). Then, users will be able to select from a variety of filters (in this case, using N, G, S, 2, 3). The resulting photo will be displayed after that filter is applied to the current photo. The user can either select another filter to be applied; or quit using the Q command (See Figure 2)



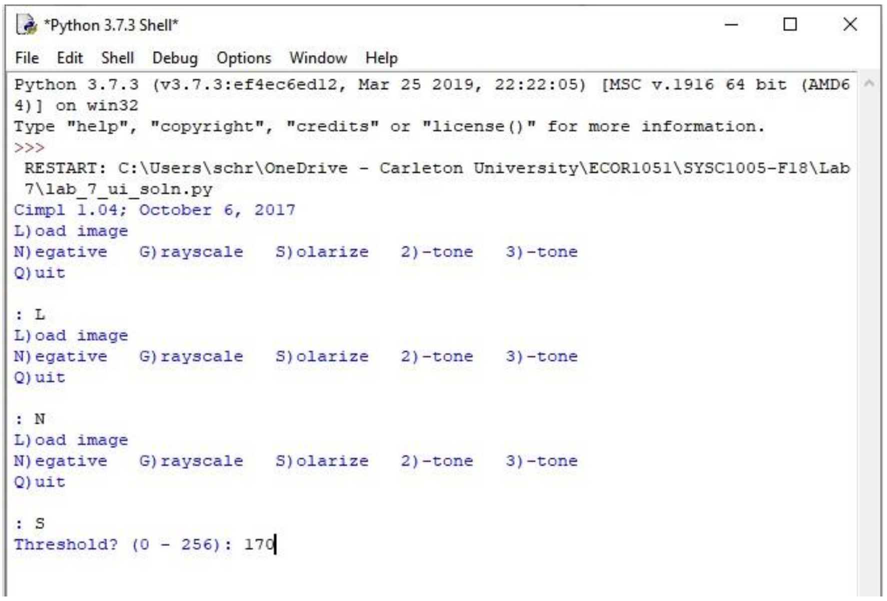
```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD6
4)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\schr\OneDrive - Carleton University\ECOR1051\SYSC1005-F18\Lab
 7\lab_7_ui_soln.py
Cimpl 1.04; October 6, 2017
L)oad image
N)egative   G)rayscale   S)olarize   2)-tone   3)-tone
Q)uit

: L
```

Name	Status	Date modified	Type	Size
__pycache__	✓	7/8/2019 1:27 PM	File folder	
Cimpl	✓	10/3/2018 7:40 PM	Python File	20 KB
great_big_c	✓	10/3/2018 7:40 PM	JPEG image	99 KB
miss_sullivan	✓	10/3/2018 7:40 PM	JPEG image	121 KB
riveter	✓	10/3/2018 7:40 PM	JPEG image	100 KB
scatter	✓	12/3/2018 11:27 PM	Python File	2 KB
scatter_soln	✓	12/3/2018 8:38 PM	Python File	2 KB

File name: riveter
All files

FIGURE 1 *SCREENSHOT OF TEXT USER INTERFACE OF PHOTO-EDITING PROGRAM (IN PROGRESS: LOADING AN IMAGE)*



```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD6
4)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\schr\OneDrive - Carleton University\ECOR1051\SYSC1005-F18\Lab
 7\lab_7_ui_soln.py
Cimpl 1.04; October 6, 2017
L)oad image
N)egative   G)rayscale   S)olarize   2)-tone   3)-tone
Q)uit

: L
L)oad image
N)egative   G)rayscale   S)olarize   2)-tone   3)-tone
Q)uit

: N
L)oad image
N)egative   G)rayscale   S)olarize   2)-tone   3)-tone
Q)uit

: S
Threshold? (0 - 256): 170
```

FIGURE 2 *SCREENSHOT OF TEXT USER INTERFACE OF PHOTO-EDITING PROGRAM - USER SELECTS N(EGATIVE) FILTER THEN S(OLIRIZE) FILTER*

Each time, the image is displayed, with the cumulative filter applied. For example, in Figure 3 below, the final image has had both the Negative and then the Solarized filter applied to it.

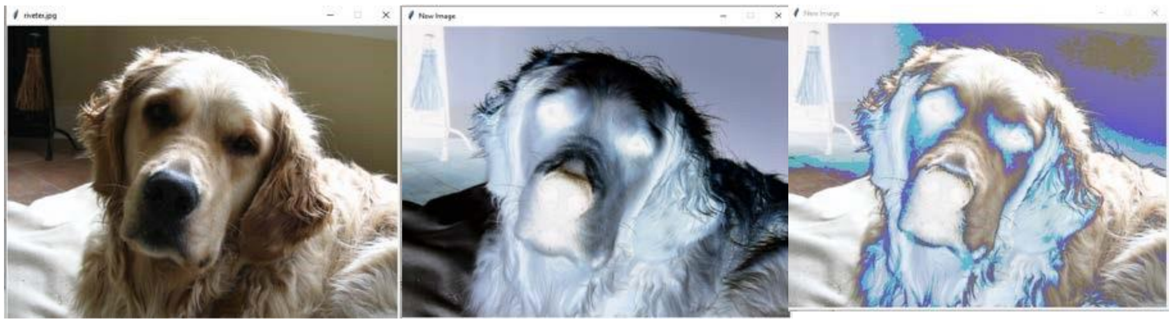


FIGURE 3 *SEQUENCE OF PHOTOS DISPLAYED (A) ORIGINAL (B) WITH NEGATIVE FILTER (C) WITH NEGATIVE + SOLARIZE FILTERS*

Only an introduction has been presented here. Full details will be presented in a structured manner that will lead your team through an *incremental*, *iterative* development cycle.

Disclaimer: Your program will **not** be exactly like this example. Here, there are 5 filters and no other features. You will implement more than 12 filters and other features.

This example is meant to help you understanding the problem and it is not meant as a marking guide

Non-Functional Requirements

1. All program deliverables must run under the lab environment described in the Course Outline.
2. All submitted files **must have the exact name required; if not, the submission will not be evaluated (zero mark)**.
3. The header of each file (the top few lines) must identify in a comment 1) the team identifier and 2) all contributing members of that team. If someone did not contribute, their name should not appear.
4. All the functions that you develop must include:
   - complete type annotations (in the function definition), and
   - a docstring containing:
     - the name of the person who wrote the function
     - a brief description of what the function does, and
     - at least one example of how we can interactively test the function from the shell.
5. All code must follow Python coding conventions.

Team Member Expectations

It is expected that all team members will participate equally in the team and that the team will meet regularly by virtual means to ensure that all project work is completed on time.

Marking Procedures

Each team will be assigned a TA who will mark their deliverables thought the course.

Some of the deliverables will be submitted by each individual of a team, while others will be submitted by only one person of the team (preferably, the team leader). Yet we recognize that sometimes the work may vary amongst individuals on a team, and therefore the marks should also vary. For that reason, the following practices will be observed:

1. Read the non-functional requirements (above) regarding the authorship of files and/or functions.
2. Every ***Submission (Assignment)*** in cuLearn will be paired with a similar individual column in the gradebook.
   a. Teams will submit their work – once, for the whole team – using the link for the Team Submission
   b. TAs will enter the individual marks for each submission using the corresponding gradebook column. To see your individual mark, you must look in the grade book (**there will be no mark entered for the Team Submission**).

Marking schemes for the various deliverables will be posted in the Milestone documents and/or in cuLearn. Check the course outline for the grade percentage assigned to the project. The weight of each deliverable is given in the "Project Grade Calculation" file posted in cuLearn. If your team is struggling, use this document to prioritize the deliverables with the heaviest weight.

Milestone Overview

The milestones will be described fully in separate documents. In this section, an overview is presented to enable you to understand the *big picture*, and the process by which software projects are managed.

Milestone 1:

- Team Formation, with a Team Contract (Expectations and Procedures)
- Problem Exploration:
    - Understanding RGB Colour Representation
    - Understanding Filters (At least one filter per team member)
- Documentation: A methodical statement of the team project, using the Project Report Template.
- Code Development: Using a unit testing framework, each individual team member will implement both a filter and its corresponding test code for one kind of image filter.

Milestone 2:

- Team Work: Delegation of tasks for the milestone; version control, schedule.
- Code Development: Implementation of some image filters
    - Each individual will prepare different image filters, as well as the unit tests for other image filters.
    - As a team, you will conduct at least two code reviews, to learn how to provide constructive feedback to your colleagues.
    - Implementation of a feature that allows you to draw curves on the image
    - The size of the total code will require organization into *modules*.
- Team Performance: Peer Evaluation with ITP (Individual and Team Performance) Metrics. The details are posted in cuLearn

(Final) Milestone 3:

- Problem Exploration:
    - Understanding an Interactive User Interface for the project
    - Understanding a Batch User Interface for the project
- Code Development:
    - Implementation of the Interactive User Interface
    - Implementation of the Batch User Interface
- Documentation:
    - Writing a readme file
- Team Performance: Peer Evaluation with ITP Metrics