CSE 8A Programming Assignment 5

Due Date: Tue. Nov 17th, 11:59 PM (PDT)

Learning goals:

- Working with 2 Dimensional lists
- Getting comfortable with nested for loops
- Learning how to represent and manipulate images as 2D lists of tuples

Logistics:

All information for the following can be found on our course syllabus.

- **Pair programming** - This programming assignment can be done individually or with a partner. Make sure to read the guide on pair programming.
- **Academic integrity** - Please adhere to all academic integrity guidelines on the syllabus.

Submission:

- You will submit the following files to Gradescope:
    - **pa5.py** - Contains your code for part 1.1, 1.2 and 1.3 (and star points, if attempted)
    - **pa5-test.py** - Contains the completed test cases as per 1.4
    - **Pa5-writeup.pdf** - Contains all written portions of the assignment
    - **Pa5-video.mp4** - Your recorded video
- You can find the template of the write-up here
- You are also required to complete the weekly reflection
- Instructions on how to submit can be found below at submission instructions.

Part 1: Implementation (Total 13 Points)

NOTE: Please do not declare global variables outside the scope of the function and use it within the function since Autograder will only call the function and does not consider the globally declared variables. Your code may thus work on your local machine but fail on Gradescope autograder.

Part 1.1. Nested for loops with Lists (2 points)

Save all of the following code in the python file named pa5.py.

Imagine you own an ice cream shop, and customers can buy ice creams of any mixture with one choice of fruits and one choice of toppings. You decided to create a menu board with every possible option included. We don't want to write down every single combination manually. So let's use nested `for` loops in order to generate a menu board. Your job will be given two lists of strings -- a list of fruits and a list of toppings. Your function has to return a list of every combination of fruits and toppings as a string. For example, when
fruits = `["apple", "banana", "strawberry"]`
toppings = `["almond", "chocolate"]`
are given as the parameters of your function, then it has to **return** the following list.

```
['apple ice cream with almond',
'apple ice cream with chocolate',
'banana ice cream with almond',
'banana ice cream with chocolate',
'strawberry ice cream with almond',
'strawberry ice cream with chocolate']
```

Function Name: *make_menu*
Parameter: *fruits* - A list of strings that contains fruits *toppings* - A list of strings that contains toppings
Return: A list of strings that indicate every possible combination in the following format `<fruit> ice cream with <topping>`
Description: Given lists of ingredients, return every possible mixture

Examples:
fruits = `["apple", "banana", "strawberry"]`
toppings = `["almond", "chocolate"]`
Return:
`['apple ice cream with almond', 'apple ice cream with chocolate', 'banana ice cream with almond', 'banana ice cream with chocolate', 'strawberry ice cream with almond', 'strawberry ice cream with chocolate']`
 	
fruits = `["apple"]`
toppings = `["almond", "chocolate", "sprinkle"]`
Return:
`['apple ice cream with almond', 'apple ice cream with chocolate', 'apple ice cream with sprinkle']`

Part 1.2: Multiplication Table (2 points)

Save all of the following code in the python file named pa5.py.

Your primary-schooler nephew asks you to make a multiplication table with positive integers. Let's use nested `for` loops and `range()` to generate a table. Your job will be given a number `num` and a 2D list `table`, and assign elements of `table` with every multiplication among `1,2,...,num`. Then, your function will return the modified 2D list `table`. When `num` is given as the largest number to multiply, a `num` x `num` 2D list will be given as `table`, with every element initialized to 0. Your function has to assign `table[i][j]` with `i*j` for every integer `1<=i<=num`, `1<=j<=num`.

To be specific, the 1st inner list of `table` is a list of multiplications between 1 and numbers from 1 to `num`. The 2nd inner list of `table` is a list of multiplications between 2 and numbers from 1 to `num`, and so on. Then, for every `i`,`j` such that `1<=i<=num` and `1<=j<=num`, the `j`-th element in the `i`-th inner list of your returned table should be the same as `i*j`.

For example, if `num=5`, then the following 5x5 table will be given to your function.
table =
```
[[0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0]]
```
Using nested for loops, your function has to modify table as
```
[[1, 2, 3, 4, 5],
 [2, 4, 6, 8, 10],
 [3, 6, 9, 12, 15],
 [4, 8, 12, 16, 20],
 [5, 10, 15, 20, 25]]
```
and return it.

Function Name: *multiplication_table*
Parameter: *num* - the largest number to be multiplicated `table` - a `num`x`num` 2D list with every element set as 0
Return: a 2D list `table` that denotes a multiplication table from 1 to num. The j-th element of the i-th inner list of `table` should be `i*j`, for any positive integers `i <= num, j <= num`.

Description:
Given a maximum number `num` and a 2D list `table`, modify and return `table` so that it denotes the table of multiplication of every number from 1 to `num`

Examples:

num = 5
table = `[[0, 0, 0, 0, 0],`
`[0, 0, 0, 0, 0],`
`[0, 0, 0, 0, 0],`
`[0, 0, 0, 0, 0],`
`[0, 0, 0, 0, 0]]`
Return:
`[[1, 2, 3, 4, 5], [2, 4, 6, 8, 10], [3, 6, 9, 12, 15], [4, 8, 12, 16, 20], [5, 10, 15, 20, 25]]`

num = 3
table = `[[0, 0, 0],`
`[0, 0, 0],`
`[0, 0, 0]]`
Return:
`[[1, 2, 3], [2, 4, 6], [3, 6, 9]]`

Part 1.3: Implementing Basic Image Filters (6 points)

Overview

In this assignment you will be implementing different image filters that modify the pixels of the image in some interesting ways. The filters you will be implementing are the same ones used in image editing applications and apps like Instagram (however, you will implement the more basic ones). While doing so, you will learn how images are stored and manipulated in a computer and you will get much more comfortable working with nested loops.

Representing Images and Colors

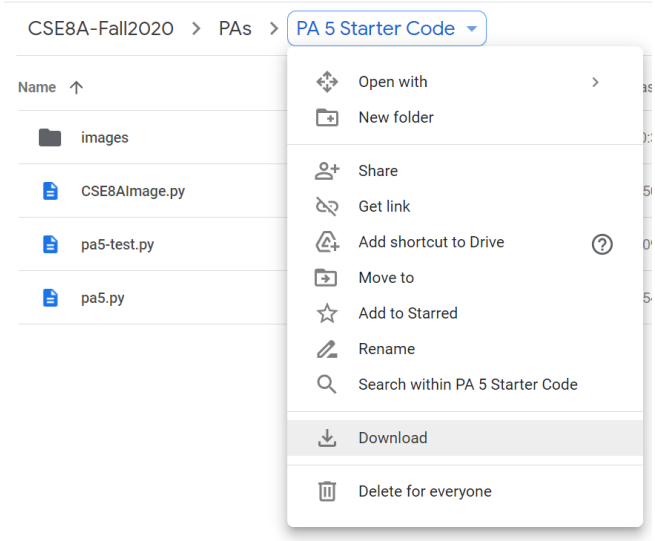
As you learnt in class, colors can be represented by three numbers describing the values of the red, green and blue color components. These numbers are in the range [0, 255]. In this assignment, we will use a 2 Dimensional list whose elements are a tuple to represent an image. We will also work with the CSE8AImage library created by the course staff which makes it easy to create new images and to query its properties.

2D Lists

One way to think of 2D lists is as a matrix. And just like with matrices, the first dimension represents the rows and the second dimension the columns. The CSE8AImage library has functions defined to query for the image height and width. The height of an image corresponds to the number of rows and the width to the number of columns. Also, remember that the top-most row is at index 0 and the bottom-most row at index height-1. Similarly, the left-most column is at index 0 and the right-most column at index width-1.

STEP A: Downloading Starter Code

To get started, first download the starter code here. Click on the dropdown at the top that says **PA5 Starter Code** and then download. This will download a zip file. Unzip it and you are good to go. You will only modify the pa5.py and pa5-test.py files. Please do **NOT** modify the contents of CSE8AImage.py file. **Please make sure you have downloaded the modified version of the CSE8AImage.py file. Check this post.**



Note: If you are not logged into Google, you will see a different interface. In this case, there will be a button on the top-right corner reading “Download All”. Click that button.

Important: Do not change the original folder structure, that is, do **NOT** rename/move files around. All the 3 files should be in the same folder with the given names.

You may add more images into the images/ folder for testing.

STEP B: Installing numpy and pillow packages

In the CSE8AImage Library provided in the starter code, you may see that we have imported packages called numpy and PIL. In order for you to be able to utilise the given library, your system should have these packages installed. Otherwise, you will get a **ModuleNotFoundError** message.

Follow these instructions carefully to install numpy and pillow packages.

1. To find the path where you need to install these packages run the below two commands (the commands that follow >>>) on the IDLE Python shell and you can see that it prints the path where you need to install the packages (In the example below the path is: 'C:\Users\winny\AppData\Local\Programs\Python\Python38')

   Commands:

   ```
   >>> import sys
   >>> print(sys.executable)
   ```

   Example:

   ```
   >>> import sys
   >>> print(sys.executable)
   C:\Users\winny\AppData\Local\Programs\Python\Python38\pythonw.exe
   >>>
   ```

2. Open Command Prompt on Windows or terminal on Mac and navigate to the path printed out in the IDLE shell using the **cd <path>** command as shown below. (Make sure that you do not navigate all the way to the last file, but only to the last directory. In the above example, we will navigate to the Python38 directory)

   Command:
   > cd <appropriate_python_path>

   Example (For Windows):

   ```
   C:\Users\winny>cd C:\Users\winny\AppData\Local\Programs\Python\Python38\
   ```

   Example (For MAC):
   When you go into idle and type the sys command and print(sys.executable), you may get something similar to below on MAC machines:
   /Library/Frameworks/Python.framework/Versions/3.8/bin/python3.8
   Make sure that you do not navigate all the way to the last file, but only to the last directory, because python3.8 is a file, not a folder. So when you are running the cd command on the **terminal** of MAC machine, run it as:
   > cd /Library/Frameworks/Python.framework/Versions/3.8/bin/

3. After navigating to the correct Python path, run the below command on the command prompt in Windows or the terminal in Mac to ensure that you have pip installed.

   Command (For Windows):
   `python -m ensurepip`

   Command (For Mac):
   `python3 -m ensurepip`

   Example:

```
C:\Users\winny\AppData\Local\Programs\Python\Python38>python -m ensurepip
```

4. Once pip is installed, you can install the numpy and pillow packages as follows using pip command in the same path as shown below. The packages will now be downloaded to your local machine if it is not present

   Command (For Windows):
   `> python -m pip install numpy`
   `> python -m pip install pillow`

   Command (For Mac):
   `> python3 -m pip install numpy`
   `> python3 -m pip install pillow`

   Example:

```
C:\Users\winny\AppData\Local\Programs\Python\Python38>python -m pip install numpy
Requirement already satisfied: numpy in c:\users\winny\appdata\local\programs\python\python38\lib\
WARNING: You are using pip version 19.3.1; however, version 20.2.4 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Users\winny\AppData\Local\Programs\Python\Python38>python -m pip install pillow
Collecting pillow
  Downloading https://files.pythonhosted.org/packages/d2/2c/9ac71068585b3d528349be6617ed963068bfcb
     |                                | 2.1MB 1.7MB/s
Installing collected packages: pillow
Successfully installed pillow-8.0.1
WARNING: You are using pip version 19.3.1; however, version 20.2.4 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

   Note that if the package has been installed previously on your machine then it will say **‘Requirement already satisfied’** otherwise it will say **‘Downloading’** and once the download is complete you will be able to use these packages. You can disregard the warning messages if any.

STEP C: Understanding CSE8AImage Documentation

Inside pa5.py you will find one import statement. That is the library you will be using to do some operations like loading images, saving images, creating images, or finding out the image properties like height and width. The documentation for the most relevant methods in the CSE8AImage library are provided below.

NOTE: You do **NOT** need to completely understand how these functions are implemented to use them. You can treat these functions as black boxes that will do the job for you in the background.

Function Name	**Input Parameters**	**Description**
load_img(filename)	**filename:** A string containing the filepath or filename where the image is	This function takes in a string containing the filepath or filename where the image is and the function **returns** the 2D list representation of the image
save_img(img, filename)	**img**: The 2D list representation of an image **filename**: A string containing the filepath or filename where the image should be saved	This function takes in a 2D list representation of an image and a string containing the filepath or filename where the image should be stored and proceeds to store the image in the given filepath and filename. The function returns None.
create_img(height, width, color)	**height**: An integer representing the height of the image to be created **width**: An integer representing the width of the image to be created **color**: A tuple containing the 3 components of the color Example: (255,255,0)	This function takes in two integers that represent the height and the width of the image to be created followed by a tuple that contains the RBG values of the color the image should be and creates a 2D list representation of the image as specified by the input parameters. The function **returns** the 2D list representation of the image thus created,
height(img)	**img**: The 2D list representation of an image	This function takes in the 2D list representation of an image and **returns** the height of the image (which is equivalent to the number of rows in a 2D list)
width(img)	**img:** The 2D list representation of an image	This function takes in the 2D list representation of an image and **returns** the

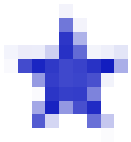
		width of the image (which is equivalent to the number of columns in a 2D list)
img_str_to_file(img, filename)	**img:** The 2D list representation of an image **filename**: A string containing the filepath or filename where the pixel values of the image should be saved	This function takes in a 2D list representation of an image and a string containing the filepath or filename where the pixel values of the image should be stored and proceeds to store the pixel values for a maximum of 20 rows and columns in the given filepath and filename. The function **returns** the pretty formatted string representation of the pixel values of the image (max 20 rows and columns).

For example:

We have provided a folder with images as part of the starter code. One of those images is a tiny 10*10 image of a blue colored star. I can load the **star.png** image using **load_img** function and then view the pixel values using the **img_str_to_file** function as follows:

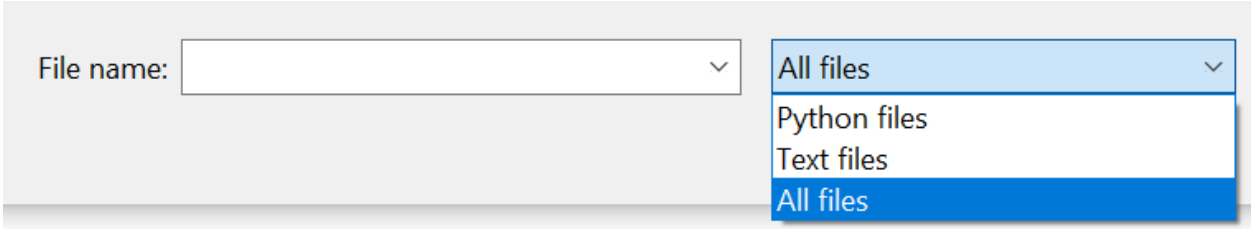
```
star = load_img("images/star.png")
img_str_to_file(star, "view_star_pixels.txt")
```

The **star.png** image looks as shown below

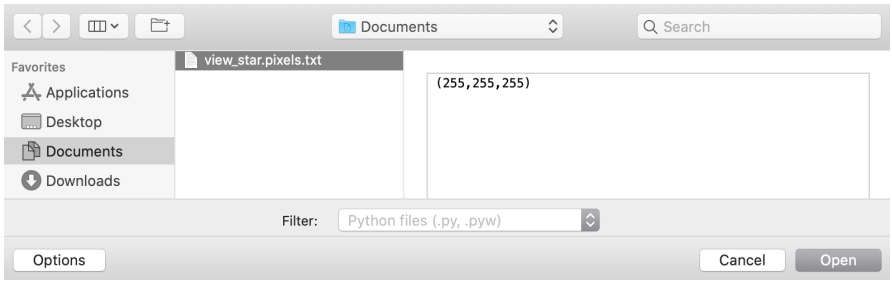


The pixel values of star.png image are written to a file called **view_star_pixels.txt** in the same folder. You can open these **.txt** pixel value files from **IDLE** using the **FILE -> OPEN**.

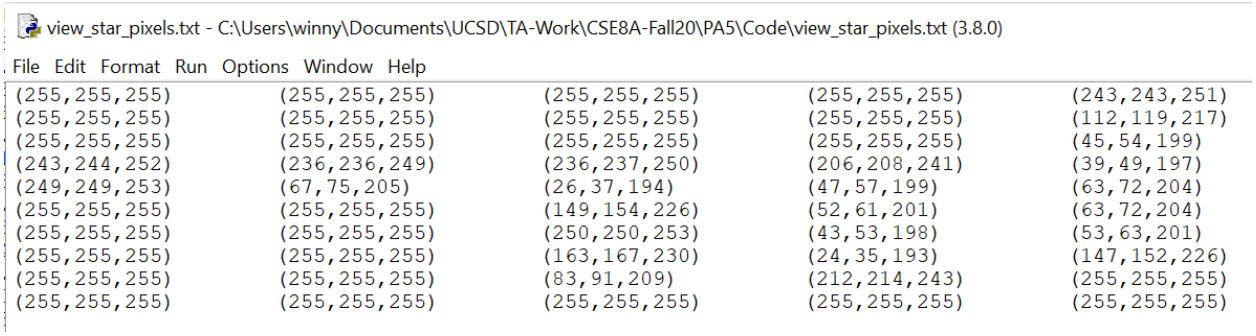
Then on Windows, choose **ALL FILES** on the right hand drop down option. You will then be able to click on the file name to open it and view its contents



For MAC, once you use **FILE -> OPEN** from **IDLE** and go to the folder, you will immediately be able to click on the file **name view_star_pixels.txt** and click Open as follows.



A part of the file contents for **view_star_pixels.txt** is shown below:



Try calling few of the functions of this library to see how it works before moving on to actually implementing the filter functions.

STEP D: Start Coding!

You will Implement and test the two mandatory image filter functions defined in the starter code.

Important: Do not change the name and parameters of the functions defined in the starter code. After implementing the methods, replace the None return value with the correct one.

For each filter explained below, the accompanying image under the description shows the result of applying the filter on the `cat.jpg` image (shown here on the right) provided under the `images/` folder in the starter code.

We suggest you implement the function and test it out on the smaller `star.png` image first before trying it out on the `cat.jpg` image.

OPTIONAL: You can find more royalty-free images like this here if you want to have fun with trying these filters on more images.

However, these images have very high resolution so you may want to reduce their size to make it easier to work with. Instructions on how to do this for Windows are here and for Mac OS here.	

Complement Filter

Function Name: *complement*
Parameter: *img* - The 2D list representation of an image
Return: A 2D list representation of complement image of of the given image
Description: This filter returns a new image where the blue and green color components of every pixel are swapped. This filter is not standard as the resulting image does not have much meaning, but can sometimes look interesting as in this example.
Example Result: 

Negative Filter

Function Name: *negative*
Parameter: *img* - The 2D list representation of an image

Return: A 2D list representation of negative version of the given image
Description: This filter converts the image into its photographic negative by returning a new image where the color of each pixel is computed as follows: Set each component of the color to be 255 minus the value of the same component in the corresponding pixel of the original image.
Example Result: 

Part 1.4 Writing Test Cases for Image Filters (3 points)

You have been given a file called **pa5-test.py** that contains these testing functions. Try to understand the test cases that are provided to you and you can add on similar test cases. You are required to write a total of 3 test cases as follows.

1. **test_complement() -** Function to test the complement function within pa5.py
   Two test cases have been given to you already. Your task here is to add **ONE** more test case with self-created images similar to the given Test Case 1.

2. **test_negative()** - Function to test the negative function within pa5.py
   One test case has been given to you already. Your task here is to add **ONE** test case with self-created images similar to the given Test Case 1. And also add **ONE** test case to get the negative of the **cat.jpg** image given in the starter code folder.

OPTIONAL: You can add on more test cases to test out the functions.

Star Points (optional):

*Save all of the following code in **pa5.py***

Read about star points on our course syllabus.

For star points, you are expected to write one additional filter other than the ones mentioned in this PA. You can refer to the internet for ideas on what filter to use. However, you cannot copy any code from the internet.

Star points will be given for explorations that are particularly creative or ambitious. Feel free to implement additional functions, or do creative things with the image. One idea is to edit an image using more complex filters such as contrast enhancement or blurring an image or a combination of some unique filters.

Part 2: Write up (2 Points)

You must answer a short question on testing in **Pa5-writeup.pdf.** A template has been provided. See submission instructions below on how to make a copy of this template to your own drive. In particular, you must provide:

A. Report Bugs and Issues
Include in your report any known bugs or issues with your program.

B. Question
Answer the following questions:

1. Write *one similarity* and *one difference* between **lists** and **tuples**.

2. What is the *order* in which the elements in the 2d list below will be printed?

```
data = [[10, 20, 30],
        [40, 50, 60],
        [70, 80, 90]]

for i in range(len(data)-1, -1, -2):
    for j in range(len(data[i])-1, -1, -1):
        print(data[j][i])
```

Your answer should be in the following ***format***:

Sample Answer:
The order in which the elements will be printed is:
`data[0][0], data[1][1], . . . , data[2][2]`

Part 3: Video (2 points)

For this part, you will create a video recording explaining the code you have written. Your video should answer the questions below. If you are working with a partner, both partners should be in the video and each partner should have some speaking point. Example of following code execution: https://youtu.be/7uw_Vi_F-dY

1. Follow the execution of the function *complement*, line by line, with an input of the **`cat.jpg`** image and show the results obtained.
2. Follow the execution of the function *negative*, line by line, with an input of **`cat.jpg`** image and show the results obtained.

The following things will be checked in your video while grading:

1. The student(s) **code is clearly visible** in the video. [Hint: Increase your font size to 18
   Windows: "Options" → "Configure Idle" → Size
   Mac: "IDLE"(in top menu bar) → "Preferences" → "Fonts/Tabs" → "Size"]
2. The student(s) clearly **answer(s) the questions**.
   a. Students submitting individually answer both questions.
   b. Students submitting in pairs each answer one of the questions.
3. Video is within **time limit** (max: **3 mins**) - **NOTE:** you have 1 additional minute for your than previous PAs.

Part 4: Weekly Reflection (1 Point)

Fill out the reflection form (link here). This weekly reflection form is not optional, it counts towards 1 point of your assignment. All students have to **individually** submit their own weekly reflection regardless if you're working with a partner. Weekly reflections are not due when the PA is due. You may submit your weekly reflection for PA 5 anytime before **11:59pm on Friday Nov 20th**.

FAQs:

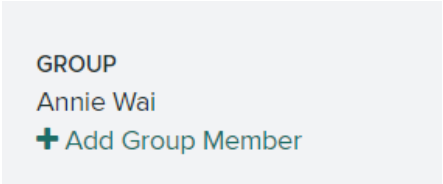
Refer to this piazza post for FAQs.

Submission Instructions

Read all instructions carefully before submitting.

- You will need to submit **pa5.py, pa5-test.py, Pa5-writeup.pdf, and Pa5-video.mp4** on Gradescope, fill out the weekly reflection, and complete the degree planning assignment.
- To copy the writeup template and export as a pdf:

    - Click here to see a copy of the assignment format.
    - Click on "File" -> "Make a copy", and you will get a local copy of this Google Doc.
    - Fill in the Google Doc, making sure you keep the headings in about the same places.
    - Once you are done, in Google Docs, click on "File"-> "Download" -> "PDF Document", which will export it to a pdf.
- To record a video on zoom:
    - Start a zoom video meeting.
    - Choose "Join with Computer Audio"
    - If you're working with a partner, invite your partner into the meeting.
    - Share your computer's screen using "Share Screen"
    - Show your code on your computer's screen
    - Click "Record" > "Record on your computer".
    - Answer the required questions.
    - Once you are done, click "stop recording" and "End meeting".
    - Save your video file on your computer and name it as Pa5-video.mp4
- Sign into Gradescope and submit all files to PA 5. You should be able to drag and drop multiple files into the upload files window. Ask a teaching staff for help if you are unsure whether you've submitted properly.
- If you are working with a partner, **only one member will need to submit the files** in Gradescope. Do not both submit the files individually to Gradescope. It will be your responsibility to ensure both members are added in Gradescope.
- To add a group member on Gradescope:
    - First submit all files to PA 5.
    - This should take you to your submissions page. Otherwise, you can view your submission by clicking on the assignment.
    - Click on **"Add Group Member"** on the top right under your name.



    - Confirm you have added your partner. You should see both you and your partner's name under "Group" in the top right after submitting.
- You may submit multiple times until the deadline. We will be grading only your latest submission. So, please make sure that your latest submission is your best version!