I/O Design

Class: Main

Fields:

Image objects - main\_image, backup\_image

Image byte data list - image\_data\_list

Methods:

open\_image\_file()

pixelator(Image obj)

save\_image\_file(Image obj)

delete\_image\_file()

reset\_image\_file()

cancel\_image\_file()

***def open\_image\_file ():***

- No parameters (called for both encoding and decoding processes)

+ Opens user’s file explorer

+ assign the image to image object class variable main\_image.

Ex.) main\_image = Image.open(image file)

+ set a second image object class variable to make copy

Ex.) backup\_image = Image.open(image file)

+ send image object to pixelator method to extract image’s byte data and convert it into a structured list. Image\_data\_list is a class variable that everyone can access

Ex.) image\_data\_list = pixelator(main\_image)

**Exceptions:**

+ Throws/Catches IOError –user selects invalid file type

* Dialog pop-up error message shown to user, user returned to main GUI. User may reattempt choosing image file.

+ Throws/Catches AttributeError – user exits out of file explorer without choosing a file

* User receives no alert; file explorer simply closes, and user is returned to main GUI. User may reattempt choosing image file

***def pixelator (image object):***

- Parameters: image object retrieved from open\_image\_file()

Used for both encoding and decoding process

+ algorithm to convert image object into byte data (list of lists? Array of lists? Need feedback)

+ arrange pixel data into data structure appropriate for encoding/decoding

- Return: image pixel data list

***def save\_image\_file (image object):***

- Parameters: image object retrieved from Zhihua/GUI

+ open user’s default native file explorer

- Within the file explorer, user can rename and choose file location where image is to be saved in computer.

- If attempting to overwrite another file:

Dialog pop-up warning “Yes/No” message asking if user wants to replace the file chosen with image.

* + - If yes:

Overwrite and save image to this location.

* + - If no:

Cancel save operation, return to file explorer

**Exceptions:**

+ Throws/Catches PermissionError –user tries to save over restricted or unauthorized file (no permission)

* Dialog pop-up error message shown to user, user returned to file explorer to try and save the image file again

+ Throws/Catches IOError –user tries to save in an unauthorized location (ex. in the C:\\Windows folder where the OS is located)

* Dialog pop-up error message shown to user, user returned to file explorer to try and save the image file again in a different location

+ Throws/Catches AttributeError – user exits out of file explorer without saving the file

* Dialog pop-up error message shown to user asking user if they are sure they want to return to GUI without saving. If yes, return user to Main GUI with encoded/decoded message ready to save again.

If no, stay in file explorer to allow user to reattempt to save file

***def reset\_image\_file ():***

+ dialog pop-up yes no alert message asking user if user wants to reset the current image to its original form

- If yes, delete main\_image variable and replace it with backup\_image. Backup image becomes the main\_image (you’re left with two copies of the original unaltered image)

Ex.) main\_image = backup\_image

- If no, do nothing, go back to main GUI

- If user exits out of dialog without selected yes or no, assume no go back to main GUI