### Name: Tradd

### As a lab, we require you to answer the following prompts. Place your responses in the boxes below.

### INITIAL DESIGN PLAN: What is a pseudocode design plan which meets the computational requirements of this lab?

|  |
| --- |
| Convert text to ASCII  Iterate through the pixels of the image.  For each pixel, replace one of the color values with an ASCII character  Then have a copy of the original image and then the edited image.  For each pixel check if the values are the same.  If they are not, add the color value that is different to a list.  Convert all of the values in the list back to text |

### SUMMARY: A brief summary description of the design and implementation, including how much your initial design plan evolved, the final result you achieved and the amount of time you spent as a programmer in accomplishing these results. This should be no more than two paragraphs.

|  |
| --- |
| The design is to take the store the text within the color values of pixels in the image. Then, the original image is used as a reference to check if the pixels have been changed. If the pixel has been changed, the values that were changed are taken back out and converted to text again. I originally wanted to take the average of the ASCII character and the color value. However, when the text was pulled back out, it was messed up. This was due to the fact that integer division was used and some values were rounded.  In the end, my program just replaces the color value with the ASCII character. I spent about 5 hours working on this program. There were some complications that I could not figure out how to handle. This caused some time set backs |

### IMPLEMENTATION: A list in bullet form of specifically what was accomplished including any challenges overcome and innovations that were not specifically required by the assignment.

|  |
| --- |
| * Encryption of text into the pixels of an image * Decryption of text from a picture using a reference image * I created a dictionary for the ASCII characters, but this didn’t include some specially formatted characters so I made a check to make sure that what is being converted is in the dictionary so it doesn’t have a run-time error |

### TESTING: A list in bulleted form of all input values used for testing. Here you should be careful to select representative input cases, including both representative typical cases as well as extreme cases.

|  |
| --- |
| * I used a large text file of just exclamation points to have a low value so I could check that my program was changing the colors * I used Act I Scene I of Hamlet as my typical test file so I could check to make sure the decryption worked correctly |

* FILES: A list in bulleted form of the names of all files submitted (source code and input, etc.)

|  |
| --- |
| * schmidtt\_csc236L1\_driver.py * schmidtt\_csc236L1\_image.ppm * schmidtt\_csc236L1\_message.txt * schmidtt\_csc236L1\_ppm |

### ERRORS: A list in bulleted form of all known errors and deficiencies

|  |
| --- |
| * \n does not work as there is not an ASCII integer representation for this * Specially formatted characters such as ”—“ |

### COMMENTS: A paragraph or so of your own comments on and reactions to the lab.

|  |
| --- |
| I thought it was a kind of confusing lab depending on how you implement the embed and decrypt functions. It can either be simple or very hard. (simple if you just replace the color’s value with the ASCII character or difficult if you take an average of the 2) |

### BIG-O on embed(): What is the big-O analysis of the method that embeds the message into your image? You may assume that the image is *n* pixels by *m* pixels and the message is of size *x*. Be careful to consider the situation when the message has more characters than there are pixels in the images, or if it has fewer.

|  |
| --- |
| O(x) |

### BIG-O on extract(): What is the big-O analysis of the method that extracts the message from the image? You may assume that the image is *n* pixels by *m* pixels and the message is of size *x*.

|  |
| --- |
| O(n\*m) |