***BPCS***

**Manual:**

1. Ensure latest version of Pillow is installed
   1. Pip install Pillow
2. Run BPCS.py
3. Choose .PNG images from image folder and pick a reasonably large vessel image
   1. Recommended a png image that does not have 1500 x 1500 dimension as it is harder for image to be compressed into the small image panel in the GUI
   2. .PNG is chosen instead of .BMP as to include more range of images
   3. .JPG cant be used as once image is compressed for stego, it will lose data
4. Choose target image that is **MUCH** smaller than vessel image to allow conjugation to execute smoother
5. Click on embed and image will be stego-fied!
6. Program will produce an image named ‘finalstego.png’ and will get this file and compress it to preview it on the image panel
7. To recover a hidden image, choose ‘finalstego.png’ and click on Recover Image
8. Program will lag slightly but just wait a while
9. Hidden image that was extracted will be saved inside main directory as ‘HiddenImageDecoded.png’
10. That’s the program

**How it works (simplified):**

1. Upon selection of image from ‘<select>’ button, image will be compressed into a 200 x 300 dimension and previewed in the image panel of the GUI
2. Dimensions and size of the original vessel image will be shown in the GUI as well
3. Same concept goes to the selection of the target Image
   1. If both target and vessel image does not pop up inside the image panel, restart your program, or select a different image
      1. Chances are, the image is too big to be compressed into the 200 x 300 format
4. Embedding of the image goes like this:
   1. Vessel image will be converted into PBC (Pure Binary Code)
   2. Then it will be converted into CGC (Cannonical Gray Code)
   3. This makes the insertion of the BPCS planes less intrusive
   4. Image will be iterated through
      1. Separates the image, bit plane wise
      2. Iterates each plane and for each complex plane located, it breaks the loop
   5. Meanwhile, target image is transformed into CGC as well and separated into 8 bit planes
   6. Once a complex plane is found inside the the vessel image, program inserts bit (target image) into the said plane and continues the loop of iteration of the vessel image plane
   7. Upon insertion, the conjugation method is called to mark the area that has been conjugated
   8. Each bit of the conjugation map represents a target image block
   9. Once conjugation is complete, program transforms image back to PBC from CGC
5. Recovery of hidden image goes like this:
   1. Program transforms vessel image to CGC from PBC
   2. Iteration is executed to loop through the vessel image planes
   3. Program checks whether a target shape exists
   4. If exists, conjugation map is printed out from the bits located inside the complex planes and recovered
   5. Bits are reassembled and recovered into an Image named ‘HiddenImagedecoded.png’
6. FIN

\*\*\* I may have some parts of the process interpreted wrongly so please feel free to edit this doc as well!

\*\*\* Many functions are actually called in the program but its too complicated to list out everything here. Do check the codes for the original comments inside as well as the additional comments ive inserted for easier understanding