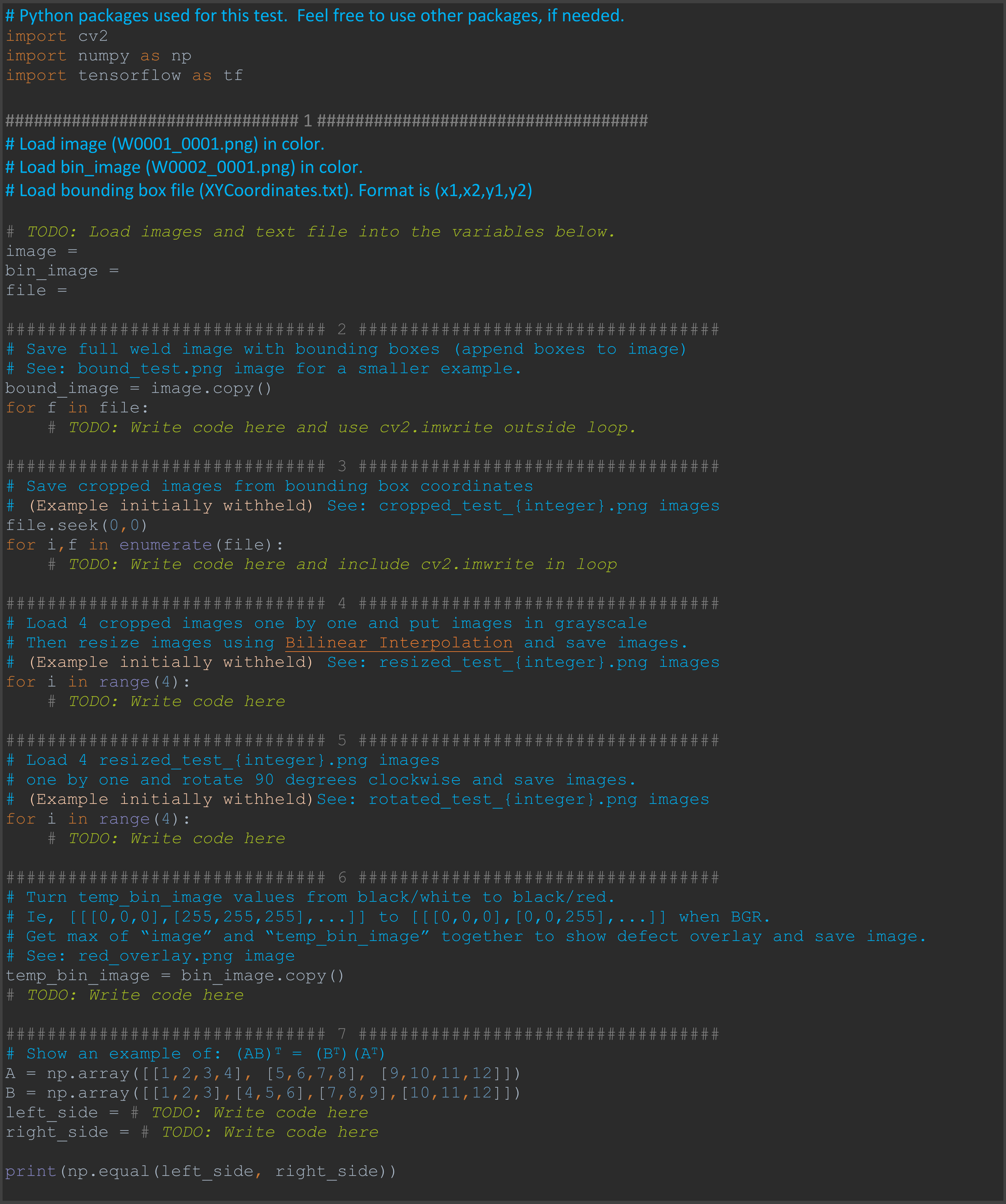
Applicant Test

Below is a list of 7 questions and a short coding project that correlate with current projects here at ORBIS. Any files referenced in the code below will be included in the folder given to you. I may withhold some of the resulting images at first, but if you're struggling, let me know and I can give you an example. If you do not understand a question, feel free to reach out to me at christophercook@orbisinc.net or by phone at 2178995643. If you do not know the answer, do not fret; you do not have to "Ace" this test. This is for us to understand your knowledge and capabilities to learn new things. On that note, feel free to use the internet for reference and supporting documents.

- For the 7 questions, I will be reviewing your approach and your resulting images.
    - You do not have to comment your code, but please save a resulting image if it asks for it.
    - You can code this in any IDE and show your code in any formatted file—ie docx, pdf, png, etc.
- For the short coding project, I will be reviewing just the code.
    - You do not have to comment your code.
    - You can code this in any IDE and show your code in any formatted document—ie docx, pdf, etc.
    - This does not have to be perfect, nor does it have to work.
    - I will be reviewing the code and seeing if you understand how to use the Python libraries correctly.

7 Coding Questions



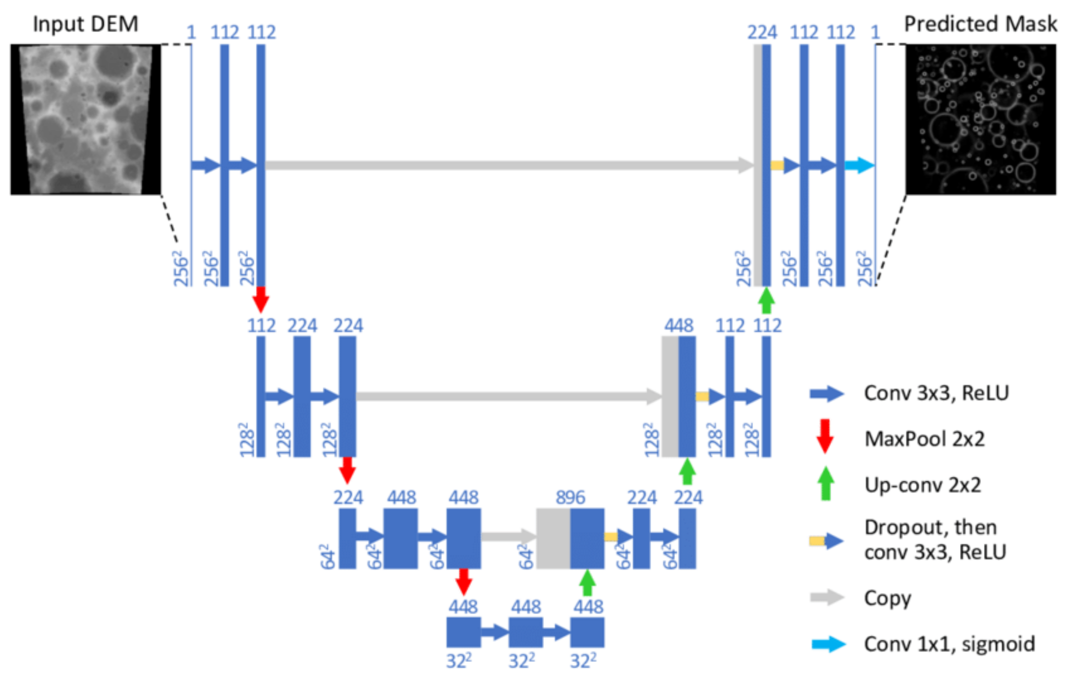
```
# Python packages used for this test.  Feel free to use other packages, if needed.
import cv2
import numpy as np
import tensorflow as tf

################################ 1 ####################################
# Load image (W0001_0001.png) in color.
# Load bin_image (W0002_0001.png) in color.
# Load bounding box file (XYCoordinates.txt). Format is (x1,x2,y1,y2)

# TODO: Load images and text file into the variables below.
image =
bin_image =
file =

############################### 2 ###################################
# Save full weld image with bounding boxes (append boxes to image)
# See: bound_test.png image for a smaller example.
bound_image = image.copy()
for f in file:
    # TODO: Write code here and use cv2.imwrite outside loop.

############################### 3 ###################################
# Save cropped images from bounding box coordinates
# (Example initially withheld) See: cropped_test_{integer}.png images
file.seek(0,0)
for i,f in enumerate(file):
    # TODO: Write code here and include cv2.imwrite in loop

############################### 4 ###################################
# Load 4 cropped images one by one and put images in grayscale
# Then resize images using Bilinear Interpolation and save images.
# (Example initially withheld) See: resized_test_{integer}.png images
for i in range(4):
    # TODO: Write code here

############################### 5 ###################################
# Load 4 resized_test_{integer}.png images
# one by one and rotate 90 degrees clockwise and save images.
# (Example initially withheld)See: rotated_test_{integer}.png images
for i in range(4):
    # TODO: Write code here

############################### 6 ###################################
# Turn temp_bin_image values from black/white to black/red.
# Ie, [[[0,0,0],[255,255,255],...]] to [[[0,0,0],[0,0,255],...]] when BGR.
# Get max of “image” and “temp_bin_image” together to show defect overlay and save image.
# See: red_overlay.png image
temp_bin_image = bin_image.copy()
# TODO: Write code here

############################### 7 ###################################
# Show an example of: (AB)^T = (B^T)(A^T)
A = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])
B = np.array([[1,2,3],[4,5,6],[7,8,9],[10,11,12]])
left_side = # TODO: Write code here
right_side = # TODO: Write code here

print(np.equal(left_side, right_side))
```

Short Coding Project

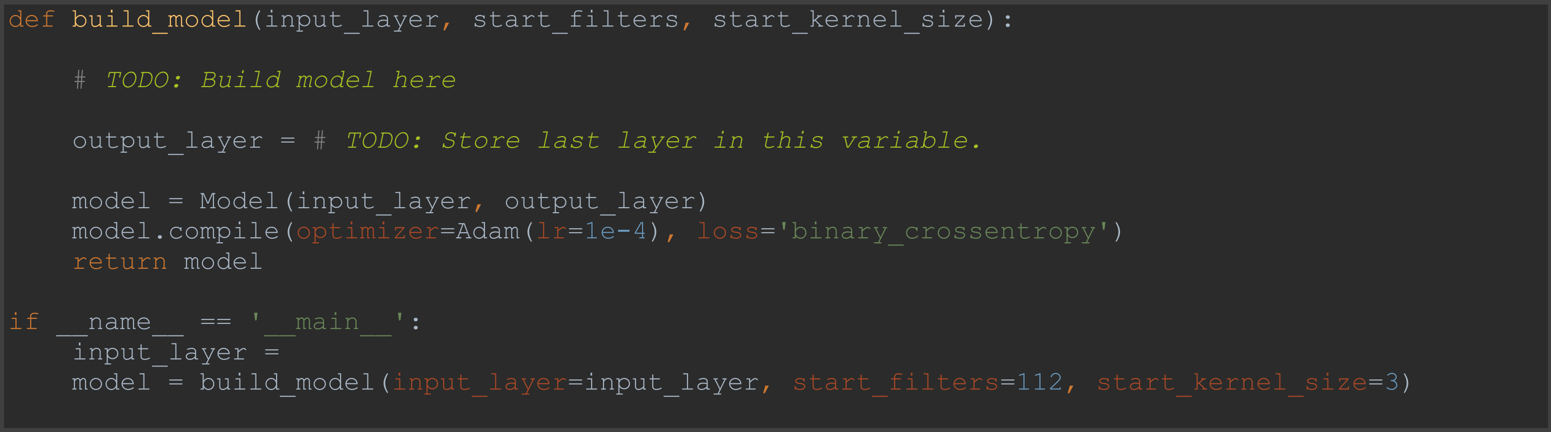
Build this simple model using Tensorflow 2.0’s Keras packages (See Image Below). If you are more comfortable with another library or group of libraries in Python, feel free to use them instead. This question does not have to be perfect, and you do not have to train it. It is used to verify that you can build a model and understand the general concepts. Feel free to use the template at the bottom of this document if you are using Tensorflow 2.0’s Keras. Again, contact me if you have any questions.

Instructions and Hints:

1. For Input:
   a. Input size should be 256x256x1—meaning grayscale images.
2. For layers:
   a. Note each layers’ sizes, in example, 256^2 = 256x256, 128^2 = 128x128, and so on.
   b. Note each layers’ number of filters, 112, 224, etc.
   c. Each layers’ kernel size = (3,3).
   d. Note each layer uses “same padding” so size won’t reduce for convolutional layers. In example, 256^2 to 256^2 and not 256^2 to 254^2
   e. For Up-Conv 2x2, feel free to use UpSampling2D or Conv2DTranspose with stride = (2,2) if using Tensorflow Keras.
   f. “Copy” is just a concatenation, so feel free to use concatenate.
      i. If you do not understand the concatenation, you can skip it.
   g. Feel free to have separate lines for Dropout and the Activation functions—like ReLU.
      i. However, any way that you can syntactically incorporate it will be fine.
3. For Output:
   a. Output size should be 256x256x1.
   b. Use sigmoid activation function on last Convolution.



U-Net Template



```python
def build_model(input_layer, start_filters, start_kernel_size):

    # TODO: Build model here

    output_layer = # TODO: Store last layer in this variable.

    model = Model(input_layer, output_layer)
    model.compile(optimizer=Adam(lr=1e-4), loss='binary_crossentropy')
    return model

if __name__ == '__main__':
    input_layer =
    model = build_model(input_layer=input_layer, start_filters=112, start_kernel_size=3)
```