Sravya's answers - Phase 1

Part 1 answers:

1. How did you verify that you are parsing the contours correctly?
    a. Added a test case with dummy data to verify all points inside and on the polygon are true
    b. Also verified visually by overlaying polygon on DICOM.
    c. In real life setting, an expert confirmation would not hurt to make sure our contour files make sense.
2. What changes did you make to the code, if any, in order to integrate it into our production code base?
    a. parse_dicom: I am returning the image array itself instead of a dict which contains array, as array is the only useful information being used elsewhere.
    b. parse_contour: I changed it to return both mask as well as polygon (clubbed poly_to_mask). It looks more consistent with parse_dicom now. But really either way is fine.
        i. Changed outline to 1, as including the polygon lines into the mask made sense, especially as we are dealing with i-contour?
        ii. Added some error checking
    c. Added few more utilities
        i. To associate contour files with correct dicom files:
            1. I am using a map with original_id(contour folder) as the key and patient_id (dicom folder) as the value.
            2. Utilities to extract original_id and dicom filename from contour filename
                a. I think '48' in the IM-0001-0048-icontour-manual.txt corresponds to the DICOM filename, as that is the only element which is changing in the contour filenames and also the numbers correspond to dicom filenames.
        ii. To visualize dicom and contours side_by_side as well as overlayed

Part 2 answers:

1. Did you change anything from the pipelines built in Parts 1 to better streamline the pipeline built in Part 2? If so, what? If not, is there anything that you can imagine changing in the future?
    a. I added a data_generator as well as another utility to process a set of contour, dicom pairs. I cleaned by previous utilities a bit.
2. How do you/did you verify that the pipeline was working correctly?
    a. Added unit tests for most functions
3. Given the pipeline you have built, can you see any deficiencies that you would change if you had more time? If not, can you think of any improvements/enhancements to the pipeline that you could build in?

a. I think we could store the preprocessed dicom and mask arrays on disk (using pickle?) to speed up data_generator

b. Add tests for error conditions and how we handle them

   i. Data folder structure is not conventional in different ways

   ii. filename formats are not conventional

c. I would do another round of polishing if I had more time

d. Add setup instructions which includes required dependencies.

e. With minor changes we can handle dicoms of sizes other than 256.