Importing Libraries

```
# Import libraries
!pip install pdf2image
!apt install poppler-utils
!apt install libtesseract-dev
!sudo apt install tesseract-ocr
!pip install pytesseract==0.3.9
!pip install tesseract

!pip install Pillow
from PIL import Image
import pytesseract
from pdf2image import convert_from_path
import cv2
import sys
import os
import csv
import numpy as np
from google.colab.patches import cv2_imshow

pytesseract.pytesseract.tesseract_cmd = r'/usr/bin/tesseract'

# Adding custom options for tesseract
# OCR Engine Mode(oem) is set to Legacy and LSTM
custom_config = r'--oem 3 --psm 6'

# Mount Google Drive
```

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Setting up tesseract-ocr-eng (1:4.00~git30-7274cfa-1.1) ...
Setting up tesseract-ocr-osd (1:4.00~git30-7274cfa-1.1) ...
Setting up tesseract-ocr (4.1.1-2.1build1) ...
Processing triggers for man-db (2.10.2-1) ...
Collecting pytesseract==0.3.9
  Downloading pytesseract-0.3.9-py2.py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.10/dist-packages (
Requirement already satisfied: Pillow>=8.0.0 in /usr/local/lib/python3.10/dist-packages (fr
Downloading pytesseract-0.3.9-py2.py3-none-any.whl (14 kB)
Installing collected packages: pytesseract
Successfully installed pytesseract-0.3.9
Collecting tesseract
  Downloading tesseract-0.1.3.tar.gz (45.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.6/45.6 MB 19.0 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: tesseract
  Building wheel for tesseract (setup.py) ... done
  Created wheel for tesseract: filename=tesseract-0.1.3-py3-none-any.whl size=45562552 sha2
  Stored in directory: /root/.cache/pip/wheels/71/c9/aa/698c579693e83fdda9ad6d6f0d8f61ed986
Successfully built tesseract
Installing collected packages: tesseract
Successfully installed tesseract-0.1.3
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (10.4.0)
Mounted at /content/gdrive
```

Functions

```
#  Some function for use later in code
# Tesseract image to boxes function on a 3 scale mage
def tessImage2Boxes(img):
    h, w, c = img.shape
```

```
    boxes = pytesseract.image_to_boxes(img, config=custom_config)
    for b in boxes.splitlines():
        b = b.split(' ')
        img = cv2.rectangle(img, (int(b[1]), h - int(b[2])), (int(b[3]), h - int(b[4])), (255, 0, 
    cv2_imshow(img)

# Tesseract image to boxes function on a grayscale image
def tessImage2Boxes_gray(img):
    h, w = img.shape
    boxes = pytesseract.image_to_boxes(img, config=custom_config)
    for b in boxes.splitlines():
        b = b.split(' ')
        img = cv2.rectangle(img, (int(b[1]), h - int(b[2])), (int(b[3]), h - int(b[4])), (255, 0, 
    cv2_imshow(img)

# Function to write jpg and text file to the
def imageToStringTextFile(imgToWrite, newFileBase, theFile):
    #print("file to write :", newFileBase + ".jpg", ", Image shape :", imgToWrite.shape)
    # get file details
    head, tail = os.path.split(theFile)
    #print("head :", head, "tail: ", tail)
    newFileName = head + "/" + newFileBase + tail[17:19]
    print("newFileName :", newFileName)

    # Write the output of the technique to a jpg
    cv2.imwrite(newFileName + ".jpg", imgToWrite)

    # Use tesseract to convert image to string - it extracts the text from the image and saves as 
    text = str(((pytesseract.image_to_string(imgToWrite, lang='eng', config=custom_config))))
    text = text.replace('-\n\n', '')

    # # Finally, write the processed text to the file.
```

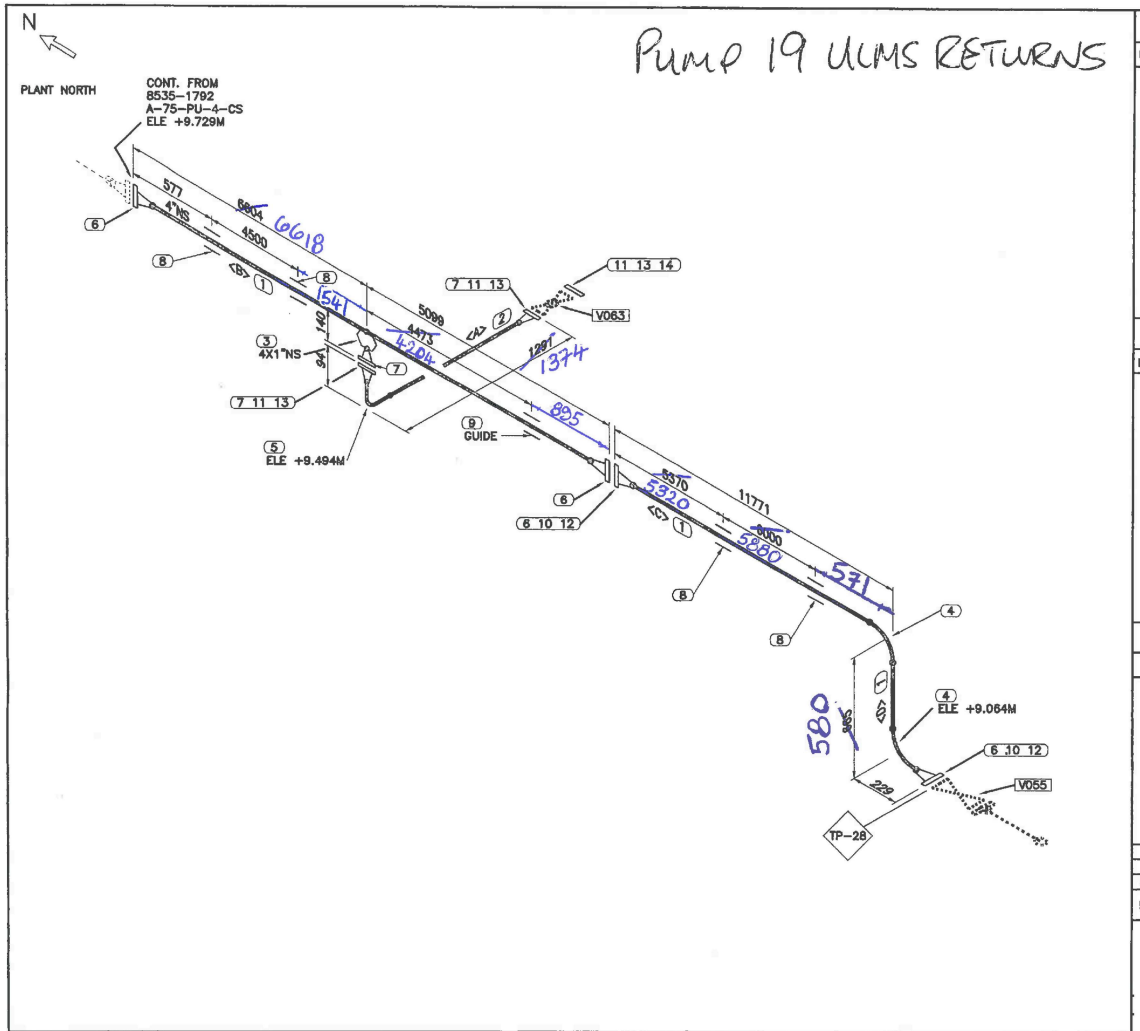
```
    with open(newFileName+".txt", 'w') as writefile:
        writefile.write(text + "\n")
```

Retrieve Files Setup

```
# File Pathname to indicate where to get of file - use google drive
afile = ('/content/gdrive/MyDrive/ComputerVision/Week 5 BOM 2')

# Read the BOM jpg here
aImg = cv2.imread(afile + '/RedactedIsometric41.jpg')
cv2_imshow(aImg)
```

PUMP 19 ULMS RETURNS



MATERIAL LIST – FABRICATION

PT.NO	SIZE	DESCRIPTION	QTY	WEIGHT (KG)
1	4	PIPE, SMLS CS SCH.40 ASTM A106 B	23.5 M	376.7
2	1	PIPE, SMLS CS SCH.80 ASTM A106 B	1.2 M	3.9
3	4X1	WELD-O-LET, BW, ASME B16.11 SCH.80 ASTM A105	1	0.2
4	4	90 DEG. LONG RADIUS ELBOW, BW, ASME B16.9 SCH.40 ASTM A234	2	7.8
5	1	90 DEG. LONG RADIUS ELBOW, BW, ASME B16.9 SCH.80 ASTM A234	1	0.2
6	4	FLANGE, WN, RF, 150 LB. ASME B16.5 ASTM A105	4	27.2
7	1	FLANGE, WN, RF, 150 LB. ASME B16.5 ASTM A105	3	3.4
8	4	PIPE SHOE PS101	4	11.6
9	4	GUIDED PIPE SHOE PS102	1	3.1

MATERIAL LIST – ERECTION

PT.NO	SIZE	DESCRIPTION	QTY	WEIGHT (KG)
10	4	GASKET, 1.6mm, 150 LB. COMPRESSED NON ASBESTOS FIBRE BS 7531	2	2.0
11	1	GASKET, 1.6mm, 150 LB. COMPRESSED NON ASBESTOS FIBRE BS 7531	3	3.0
12	5/8	(8) BOLT SET, 150LB ASTM A193 Gr.B7 BOLTS/ASTM A194 Gr.2H NUTS (90 MM LG)	2	3.1
13	1/2	(4) BOLT SET, 150LB ASTM A193 Gr.B7 BOLTS/ASTM A194 Gr.2H NUTS (65 MM LG)	3	1.6
14	1	BLIND FLANGE, RF, 150 LB. ASME B16.5 ASTM A105	1	1.0

CUTTING LIST

CUT PIECE	SIZE	LENGTH (MM)	END PREP-1	END-PREP-2
A	1"	1198	BEVEL	BEVEL
B	4"	11551	BEVEL	BEVEL
C	4"	11541	BEVEL	BEVEL
D	4"	360	BEVEL	BEVEL

REV	DESCRIPTION	BY	DATE	CHK	REV	DESCRIPTION	BY	DATE	CHK
C1	ISSUED FOR CONSTRUCTION	BS	20/05/16	LJP					

Extract Region of Interest

```
aImg.shape
```

```
(5841, 8264, 3)
```

```
# Define the region of interest - ensure this is as accurate as possible
# Coordinates obtained from MS Paint

# File Pathname to indicate where to get of file - use google drive
afile = ('/content/gdrive/MyDrive/ComputerVision/Week 5 BOM 2')

# Read the BOM jpg here
aImg = cv2.imread(afile + '/RedactedIsometric41.jpg')

# I had an error here finding ROI. After several attempts I managed to get a decent result.
# The x and y values needed to be offset by minus one. Solutioin found online.
```

```
height, width, _ = aImg.shape

row_start = max(0, min(585, height - 1))
row_end = max(0, min(3793, height))
col_start = max(0, min(5328, width - 1))
col_end = max(0, min(7664, width))

aImg = aImg[row_start:row_end, col_start:col_end]

# Call Function to write jpg and text file
imageToStringTextFile(aImg, "BOM_ROI_41", afile)
# cv2_imshow(aImg)

# Use tesseract to display image of boxes surrounding the text found. Good visual display to see h
# This can be commented out as it doesn't always need to run
# tessImage2Boxes(aImg)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_ROI_41
```

Convert To GrayScale

```
img1grayscale = aImg.copy()
# Convert to grayscale
img1grayscale = cv2.cvtColor(img1grayscale,cv2.COLOR_BGR2GRAY)
```

Binary Threshold

```
# Using Simple threshold to improve the ratio rate significantly across all images
# Highest final ratio achieved with Binary ToZero as follows.
```

```
ret, imgBin = cv2.threshold(img1grayscale,127,255,cv2.THRESH_TOZERO_INV) # TOZERO_INV almost twice

# Call Function to write jpg and text file
imageToStringTextFile(imgBin, "BOM_BIN_41", afile)

# Use tesseract to display image of boxes surrounding the text found. Good visual display to see h
# This can be commented out as it doesn't always need to run
#tessImage2Boxes_gray(imgBin)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_BIN_41
```

Erosion

```
# Inverse Threshold used since Erosion works better with a dark background.
ret, imgBinInv = cv2.threshold(img1grayscale,127,255,cv2.THRESH_BINARY_INV)

# create kernel
kernel = np.ones((3,3),np.uint8)

# Apply erosion to the image
# Spend time finding the correct balance of variables(kernel, iterations)
imgErode = cv2.erode(imgBinInv,kernel,iterations = 2) # inversethresh used here (imgBin) as erosio
#cv2_imshow(imgErode)

# Function to write jpg and text file
imageToStringTextFile(imgErode, "BOM_ERO_41", afile)

# Use tesseract to display image of boxes surrounding the text found. Good visual display to see h
# This can be commented out as it doesn't always need to run
# tessImage2Boxes_gray(imgErode)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_ERO_41
```

```
imgContour = img1grayscale.copy()
cv2_imshow(imgContour)
```

MATERIAL LIST – FABRICATION

PT.NO	SIZE	DESCRIPTION	QTY	WEIGHT (KG)
1	4	PIPE, SMLS CS SCH.40 ASTM A106 B	23.5 M	376.7
2	1	PIPE, SMLS CS SCH.80 ASTM A106 B	1.2 M	3.9
3	4X1	WELD-O-LET, BW, ASME B16.11 SCH.80 ASTM A105	1	0.2
4	4	90 DEG. LONG RADIUS ELBOW, BW, ASME B16.9 SCH.40 ASTM A234	2	7.8
5	1	90 DEG. LONG RADIUS ELBOW, BW, ASME B16.9 SCH.80 ASTM A234	1	0.2
6	4	FLANGE, WN, RF, 150 LB. ASME B16.5 ASTM A105	4	27.2
7	1	FLANGE, WN, RF, 150 LB. ASME B16.5 ASTM A105	3	3.4
8	4	PIPE SHOE PS101	4	11.6
9	4	GUIDED PIPE SHOE PS102	1	3.1

MATERIAL LIST – ERECTION

PT.NO	SIZE	DESCRIPTION	QTY	WEIGHT (KG)
10	4	GASKET, 1.6mm, 150 LB. COMPRESSED NON ASBESTOS FIBRE BS 7531	2	2.0
11	1	GASKET, 1.6mm, 150 LB. COMPRESSED NON ASBESTOS FIBRE BS 7531	3	3.0
12	5/8	(8) BOLT SET, 150LB ASTM A193 Gr.B7 BOLTS/ASTM A194 Gr.2H NUTS (90 MM LG)	2	3.1
13	1/2	(4) BOLT SET, 150LB ASTM A193 Gr.B7 BOLTS/ASTM A194	3	1.6

		GF.2H NUTS (65 MM LG)		
14	1	BLIND FLANGE, RF, 150 LB. ASME B16.5 ASTM A105	1	1.0

CUTTING LIST

CUT PIECE	SIZE	LENGTH (MM)	END PREP-1	END-PREP-2
A	1"	1198	BEVEL	BEVEL
B	4"	11551	BEVEL	BEVEL
C	4"	11541	BEVEL	BEVEL
D	4"	360	BEVEL	BEVEL

Contouring - Horizontal and Vertical lines

```
# # making a copy of the image here as findContours might make changes to original image
#imgContour = img1grayscale.copy()

# # Image has to be be in grayscal for the morphologyEx operations to work
#gray = cv2.cvtColor(imgContour, cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(imgContour,127,255,cv2.THRESH_BINARY)

# Horizontal Kernel code here: a horizontal kernel will help to detect all the horizontal line fro
horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,2) ) # This kernel yielded best r
detect_horizontal = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, horizontal_kernel, iterations=2) # 2 
cnts = cv2.findContours(detect_horizontal, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
# If third column value is NOT equal to -1 than its internal contouring
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
for c in cnts:
  # Draw the contour
  cv2.drawContours(imgContour, [c], -1, (0,0,0), 1) # Values change to fix fully white image

# Vertical Kernel code here : A verticle kernel which will detect all the verticle lines from the 
vertical_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (2,3)) # This kernel yielded best resu
detect_vertical = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, vertical_kernel, iterations=3) # 3 itera
cnts = cv2.findContours(detect_vertical, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
# If third column value is NOT equal to -1 than its internal contouring
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
for c in cnts:
  # Draw the contour
  cv2.drawContours(imgContour, [c], -1, (0,0,0), 1) # Values change to fix fully white image

# Function to write jpg and text file
```

```
imageToStringTextFile(imgContour, "BOM_HOR_VER_41", afile)

# Use tesseract to display image of boxes surrounding the text found
#tessImage2Boxes_gray(imgContour)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_HOR_VER_41
```

Erosion and Contouring

```
# apply contouring to the eroded image

# # making a copy of the image here as findContours might make changes to original image
result = imgErode.copy()
# # Image has to be be in grayscal for the morphologyEx operations to work
ret, thresh = cv2.threshold(result,127,255,cv2.THRESH_BINARY)

# Horizontal Kernel
# These values will not give the best result - you are expected to explore different values
# # This kernel yeilded best result, trial and error
horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (1,3) ) # This kernel yielded best res
horizontal_mask = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, horizontal_kernel, iterations=1)
cnts = cv2.findContours(horizontal_mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
for c in cnts:
  cv2.drawContours(result, [c], -1, (0,0,0), 1)

# Vertical Kernel
# These values will not give the best result - you are expected to explore different values
# # This kernel yeilded best result, trial and error
vertical_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,2)) # This kernel yielded best result
detect_vertical = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, vertical_kernel, iterations=2) # 2 iterat
```

```
cnts = cv2.findContours(detect_vertical, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
for c in cnts:
  cv2.drawContours(result, [c], -1, (0,0,0), 1)

# Function to write jpg and text file
imageToStringTextFile(result, "BOM_ERO_HOR_VER_41", afile)

# Use tesseract to display image of boxes surrounding the text found
#tessImage2Boxes_gray(result)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_ERO_HOR_VER_41
```

Contouring and then Erosion you are here

```
kernel = np.ones((2,3),np.uint8)

imgContour2 = imgContour.copy()

contErode =  cv2.erode(imgContour2,kernel,iterations = 1)

# Function to write jpg and text file
imageToStringTextFile(contErode, "BOM_HOR_VER_ERO_41", afile)

# Use tesseract to display image of boxes surrounding the text found. Good visual display to see h
# This can be commented out as it doesn't always need to run
#tessImage2Boxes_gray(contErode)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_HOR_VER_ERO_41
```

Blurring

```
imgBlur = cv2.imread(afile + '/RedactedIsometric41.jpg')

height, width, _ = imgBlur.shape

row_start = max(0, min(585, height - 1))
row_end = max(0, min(3793, height))
col_start = max(0, min(5328, width - 1))
col_end = max(0, min(7664, width))

imgBlur = imgBlur[row_start:row_end, col_start:col_end]

#Highest end number attained with median blur
imgBlur = cv2.GaussianBlur(imgBlur,(3,3), 5) # Changed to diff (oddnumbered) parameters. 3=best

# Function to write jpg and text file
imageToStringTextFile(imgBlur, "BOM_BLUR_41", afile)

# Use tesseract to display image of boxes surrounding the text found. Good visual display to see h
# This can be commented out as it doesn't always need to run
# tessImage2Boxes(imgBlur)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_BLUR_41
```

Contouring and then Blurring

```
# Applying a blur can help reduce noise, again making it easier for Tesseract to correctly OCR the
```

```
imgContBlur = imgContour.copy()
# use the image that has contouring applied, now apply blurring
imgContBlur = cv2.blur(imgContBlur, (5,5))

# Function to write jpg and text file
imageToStringTextFile(imgContBlur, "BOM_HOR_VER_BLUR_41", afile)

# Use tesseract to display image of boxes surrounding the text found. Good visual display to see h
# This can be commented out as it doesn't always need to run
# tessImage2Boxes_gray(imgContBlur)
```

```
newFileName : /content/gdrive/MyDrive/ComputerVision/BOM_HOR_VER_BLUR_41
```

```
# Comparing Ground truth with the ROI
from difflib import SequenceMatcher

ratioResults = []


# Read text files
with open('/content/gdrive/MyDrive/ComputerVision/Week 5 BOM 2/GroundTruth-BOM41.txt', 'r') as f:
    groundTruth = f.read()

with open('/content/gdrive/MyDrive/ComputerVision/BOM_ROI_41.txt', 'r') as f:
    roi = f.read()

with open('/content/gdrive/MyDrive/ComputerVision/BOM_BIN_41.txt', 'r') as f:
    bin = f.read()

with open('/content/gdrive/MyDrive/ComputerVision/BOM_ERO_41.txt', 'r') as f:
    erode = f.read()
```

```
with open('/content/gdrive/MyDrive/ComputerVision/BOM_HOR_VER_41.txt', 'r') as f:
    hor_ver = f.read()

with open('/content/gdrive/MyDrive/ComputerVision/BOM_ERO_HOR_VER_41.txt', 'r') as f:
    erode_hor_ver = f.read()

with open('/content/gdrive/MyDrive/ComputerVision/BOM_HOR_VER_ERO_41.txt', 'r') as f:
    hor_ver_erode = f.read()

with open('/content/gdrive/MyDrive/ComputerVision/BOM_BLUR_41.txt', 'r') as f:
    blur = f.read()
```