# Algorithm 1:

Input: Dataset Path

Output: Preprocessed Train, Test, and Validation data

1. Begin.
2. Store the dataset directory path into a variable.
3. Store the sub-directory paths for train, test, and validation folders into different variables.
4. Generate augmented images.
5. Change the size of the images to 224x224.
6. Save the modified images as training\_set, test\_set, and validation\_set.
7. End.

Then, in algorithm 2, we structure our custom ResNet-50 Convolutional Neural Network and train and validate it using the Train and Validation set.

# Algorithm 2:

Input: Train and Validation Set

Output: Trained Custom ResNet-50 model

1. Begin.
2. Initialize an empty sequential model as resnet\_model.
3. Download the ResNet-50 pre-trained model.
4. Assign as pretrained\_model and initialize with parameters - include\_top=False, input\_shape=(224,224,3), pooling=avg, classes=3, weights=imagenet.
5. For every layer in the pre-trained model, set layer.trainable=False.
6. Add the modified pretrained\_model to the resnet\_model.
7. Add a flattening layer to the sequential model.
8. Add a Dense layer of 256 neurons with ReLU activation function.
9. As the final layer, add a dense layer of 3 neurons with SoftMax activation function.
10. Compile resnet\_model with – Adam optimizer with learning\_rate=0.001, binary\_crossentropy as the loss function, and accuracy for the metrics parameter.
11. For callback functions, set ModelCheckpoint with model save location and save\_best\_only=True.
12. Set EarlyStopping with monitor=val\_accuracy, mode=max, patience=5, restore\_best\_weights=True.
13. Run fit\_generator() on the resnet\_model as ‘history’ with train\_set, validation\_data=val\_set, epochs=50, and callbacks with EarlyStopping and ModelCheckoint.
14. Set pretrained\_model.trainable = True.
15. Compile resnet\_model with optimizer Adam with learning\_rate=1e-5, loss=binary\_crossentropy, metrics=accuracy.
16. Set EarlyStopping with monitor=val\_accuracy, mode=max, patience=10, restore\_best\_weights=True.
17. Run fit\_generator() on the resnet\_model as ‘history’ with train\_set, validation\_data=val\_set, epochs=50, and callbacks with EarlyStopping and ModelCheckoint.
18. Graph the changes in accuracy and loss over time for ‘history’.
19. Finish.

Finally, we test our trained custom ResNet-50 model to find out its true capability.