Task1:

**Implementing Basic Convolution Operations**

* **Objective:** Implement a 2D convolution manually and visualize the output.
* **Steps:**
  + Load a grayscale image and define a 3x3 kernel (e.g., for edge detection).
  + Implement the convolution operation manually using nested loops.
  + Display the original and convolved images side-by-side.
  + **Expected Outcome:** Students understand the mechanics of convolution by manually coding it.

Task2:

1. **Objective:** Experiment with different padding and stride values.
2. **Steps:**
3. Define convolution with same and valid padding, as well as different stride values.
4. Apply the convolution on the same image with each configuration and observe the output size.
5. Record and discuss the changes in output dimensions.

**Expected Outcome:** Students see the effect of padding and stride on output dimensions.

Task3:

1. **Objective:** Perform convolution on RGB images to explore multi-channel input.
2. **Steps:**
3. Load an RGB image and define a 3x3 kernel.
4. Implement convolution for each color channel individually, then combine channels.
5. Display the processed image.

**Expected Outcome:** Students understand how convolution operates on multi-channel images.

Task4:

1. **Objective:** Implement max pooling and average pooling on a feature map.
2. **Steps:** Generate a feature map (or use a previously convolved image).
3. Write functions for max pooling and average pooling with a 2x2 filter.
4. Apply both pooling types and observe the effects on the output size.

**Expected Outcome:** Students visualize how pooling reduces feature map size and retains essential features.

Task5:

* Build and train the LeNet model on the MNIST dataset.
* **Steps:**
  + Define the LeNet model architecture using a deep learning library (e.g., PyTorch or TensorFlow).
  + Train the model on the MNIST dataset.
  + Evaluate the model and visualize sample predictions.
* **Expected Outcome:** Students apply their convolutional knowledge to build, train, and evaluate a classic CNN model.