這段程式碼實作了兩種生成對抗網絡（GAN）模型：傳統的 DCGAN（Deep Convolutional GAN）和改進的 WGAN-GP（Wasserstein GAN with Gradient Penalty）。以下是對這段程式碼的總結：

主要步驟：

1. **準備數據集**：

   - 使用 PyTorch 的`torchvision`載入 MNIST 數據集，並進行標準化處理。

2. **建立 DCGAN 模型**：

   - **生成器（Generator）**：使用反卷積層來生成圖像，並使用批量正則化（Batch Normalization）和 LeakyReLU 激活函數。

   - **判別器（Discriminator）**：使用卷積層來辨識圖像，並使用批量正則化（Batch Normalization）和 LeakyReLU 激活函數。

3. **訓練 DCGAN 模型**：

   - **訓練判別器**：包含對真實數據和生成數據的損失計算。

   - **訓練生成器**：通過最小化判別器對生成數據的預測來提升生成器性能。

   - 訓練過程中定期儲存生成的圖像以觀察生成效果。

4. **建立 WGAN-GP 模型**：

   - **生成器（Generator）**：類似於 DCGAN，但使用實例正則化（Instance Normalization）代替批量正則化。

   - **判別器（Discriminator）**：改為 WGAN 的形式，輸出不再是概率值，而是實數值。

- **梯度懲罰（Gradient Penalty）**：用於提高 WGAN 的穩定性，通過強制梯度的 L2 範數為 1 來實現。

5. **訓練 WGAN-GP 模型**：

- **訓練判別器**：計算真實數據和生成數據的 WGAN 損失，並加入梯度懲罰。

- **訓練生成器**：通過最小化生成數據的判別器輸出來提升生成器性能。

結果可視化：

- 在每個訓練周期（epoch）後，將生成的圖像進行可視化，展示不同訓練階段的生成效果。

這段程式碼展示了如何使用 DCGAN 和 WGAN-GP 模型來生成圖像，並且比較了這兩種模型在圖像生成質量上的差異。