```python

import tensorflow as tf

from tensorflow.keras.datasets import mnist

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

from tensorflow.keras.optimizers import Adam

# Load MNIST dataset

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

# Preprocess the data

x\_train = x\_train.reshape(-1, 28, 28, 1) / 255.0

x\_test = x\_test.reshape(-1, 28, 28, 1) / 255.0

y\_train = tf.keras.utils.to\_categorical(y\_train, num\_classes=10)

y\_test = tf.keras.utils.to\_categorical(y\_test, num\_classes=10)

# CNN Architecture 1

model\_1 = Sequential([

Conv2D(32, (3, 3), activation='relu', input\_shape=(28, 28, 1)),

MaxPooling2D((2, 2)),

Flatten(),

Dense(256, activation='relu'),

Dense(10, activation='softmax')

])

# Compile and train the model

model\_1.compile(optimizer=Adam(), loss='categorical\_crossentropy', metrics=['accuracy'])

model\_1.fit(x\_train, y\_train, batch\_size=128, epochs=10, validation\_data=(x\_test, y\_test))

# CNN Architecture 2

model\_2 = Sequential([

Conv2D(16, (3, 3), activation='relu', input\_shape=(28, 28, 1)),

MaxPooling2D((2, 2)),

Conv2D(32, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Flatten(),

Dense(256, activation='relu'),

Dense(10, activation='softmax')

])

# Compile and train the model

model\_2.compile(optimizer=Adam(), loss='categorical\_crossentropy', metrics=['accuracy'])

model\_2.fit(x\_train, y\_train, batch\_size=128, epochs=10, validation\_data=(x\_test, y\_test))

# CNN Architecture 3

model\_3 = Sequential([

Conv2D(8, (3, 3), activation='relu', input\_shape=(28, 28, 1)),

MaxPooling2D((2, 2)),

Conv2D(16, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Conv2D(32, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Flatten(),

Dense(256, activation='relu'),

Dense(10, activation='softmax')

])

# Compile and train the model

model\_3.compile(optimizer=Adam(), loss='categorical\_crossentropy', metrics=['accuracy'])

model\_3.fit(x\_train, y\_train, batch\_size=128, epochs=10, validation\_data=(x\_test, y\_test))

```