```
"""
This code implements a deep neural network model using TensorFlow, a
popular library for machine learning. The purpose of the model is to learn
the relationship between temperatures in degrees Celsius and Fahrenheit.
To achieve this, the code employs a deep neural network composed of
several hidden layers.
"""
```

```
import tensorflow as tf
import numpy as np
```

```
# Training data
celsius = np.array([-40, -10, 0, 8, 15, 22, 38], dtype=float)
fahrenheit = np.array([-40, 13, 32, 46, 59, 72, 100], dtype=float)
```

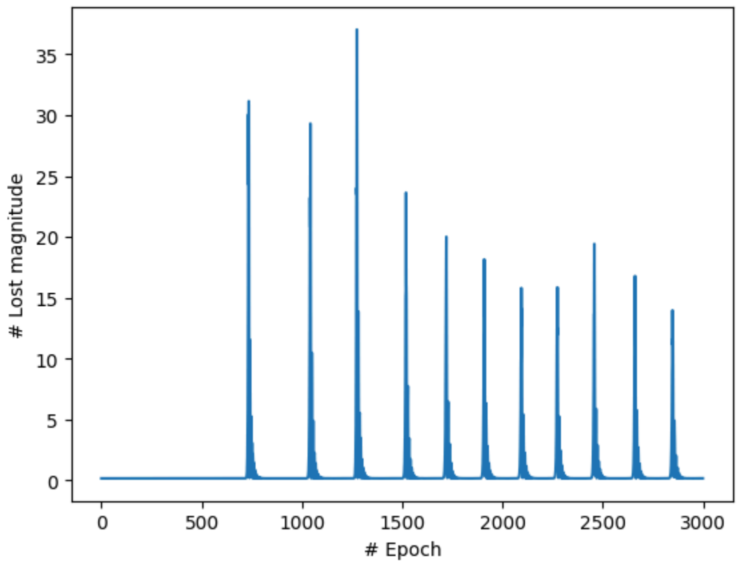
```
# Neural network configuration
hidden1 = tf.keras.layers.Dense(units=3, input_shape=[1])
hidden2 = tf.keras.layers.Dense(units=4)
hidden3 = tf.keras.layers.Dense(units=5)
hidden4 = tf.keras.layers.Dense(units=4)
hidden5 = tf.keras.layers.Dense(units=3)
exit = tf.keras.layers.Dense(units=1)
model = tf.keras.Sequential([hidden1, hidden2, hidden3, hidden4, hidden5, exit])
```

```
# Model compilation
model.compile(
    optimizer=tf.keras.optimizers.Adam(0.1),
    loss='mean_squared_error'
)
```

```
# Training process
print("Training...")
nztm = model.fit(celsius, fahrenheit, epochs=3000, verbose=False)
print("Trained!")
```

```
# Loss visualization during training
import matplotlib.pyplot as plt
plt.xlabel("# Epoch")
plt.ylabel("# Lost magnitude")
plt.plot(nztm.history["loss"])
```

```
[<matplotlib.lines.Line2D at 0x7b2f7431d570>]
```



```
# Prediction based on user input
data = float(input("Enter the degrees Celsius to convert: "))

print("Let's make a prediction!")
result = model.predict([[data]])
print("The result is " +  str(result) + " fahrenheit!")
```

```
Enter the degrees Celsius to convert: 100
Let's make a prediction!
1/1 [==============================] - 0s 103ms/step
The result is [[212.30501]] fahrenheit!
```