In [53]:
```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from sklearn.metrics import mean_squared_error
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.model_selection import train_test_split
```

In [54]:
```
df = np.genfromtxt('/content/Fertility_Diagnosis.txt', delimiter=',')
```

In [55]:
```
x = df[:, :-1]
y = df[:, -1].reshape(-1,1)
print(x.shape, y.shape, np.unique(y))

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state= 0)
```

```
(100, 9) (100, 1) [0. 1.]
```

In [56]:
```
input_size = x.shape[1]
output_size = y.shape[1]
```

In [57]:
```
error_tolerance = 0.05
learning_rate = 0.001

training_errors = []
testing_errors = []
```

In [58]:
```
for num_neurons in range(1, 10):
    print(f"Training with {num_neurons} neurons in hidden layer")

    weights_input_hidden_new = np.random.random((input_size, num_neurons))
    weights_hidden_output_new = np.random.random((num_neurons, output_size))

    for epoch in range(1, 1000001):
```

```
        layer_input_new = x_train
        layer_hidden_new = 1 / (1 + np.exp(-(np.dot(layer_input_new, weights_input_hidden_new))))
        layer_output_new = 1 / (1 + np.exp(-(np.dot(layer_hidden_new, weights_hidden_output_new))))

        error_new = y_train.reshape(-1, 1) - layer_output_new

        # Derivative of sigmoid: sigmoid(x) * (1 - sigmoid(x))
        layer_output_derivative_new = error_new * (layer_output_new * (1 - layer_output_new))
        layer_hidden_error_new = layer_output_derivative_new.dot(weights_hidden_output_new.T)
        layer_hidden_derivative_new = layer_hidden_error_new * (layer_hidden_new * (1 - layer_hidden_new))

        # Update weights
        weights_hidden_output_new += layer_hidden_new.T.dot(layer_output_derivative_new) * 0.001
        weights_input_hidden_new += layer_input_new.T.dot(layer_hidden_derivative_new) * 0.001

        if epoch % 100000 == 0:
            print(f"Epoch {epoch}, Corresponding Error: {np.mean(np.abs(error_new))}")

        if np.mean(np.abs(error_new)) < error_tolerance:
            print(f"Converged at epoch {epoch}, Error: {np.mean(np.abs(error_new))} \n")
            break

        if epoch == 1000000:
            print("Model did not converge \n")

    # Training results
    layer_input_new = x_train
    layer_hidden_new = 1 / (1 + np.exp(-(np.dot(layer_input_new, weights_input_hidden_new))))
    layer_output_new = 1 / (1 + np.exp(-(np.dot(layer_hidden_new, weights_hidden_output_new))))

    train_error_new = np.mean(np.square(y_train.reshape(-1, 1) - layer_output_new))
    training_errors.append(train_error_new)
    print(f"Training Mean Squared Error: {train_error_new}")

    # Testing results
    layer_input_new = x_test
    layer_hidden_new = 1 / (1 + np.exp(-(np.dot(layer_input_new, weights_input_hidden_new))))
    layer_output_new = 1 / (1 + np.exp(-(np.dot(layer_hidden_new, weights_hidden_output_new))))
```

```
    test_error_new = np.mean(np.square(y_test.reshape(-1, 1) - layer_output_new))
    testing_errors.append(test_error_new)
    print(f"Testing Mean Squared Error: {test_error_new}")
```

```
Training with 1 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.1675926727817341
Epoch 200000, Corresponding Error: 0.16589119596640614
Epoch 300000, Corresponding Error: 0.1643946091793576
Epoch 400000, Corresponding Error: 0.1632744379663421
Epoch 500000, Corresponding Error: 0.16240084393899115
Epoch 600000, Corresponding Error: 0.16169849438579972
Epoch 700000, Corresponding Error: 0.16112063207138286
Epoch 800000, Corresponding Error: 0.16063508743233154
Epoch 900000, Corresponding Error: 0.160217750488757
Epoch 1000000, Corresponding Error: 0.15984892722464875
Model did not converge

Training Mean Squared Error: 0.08151322511694374
Testing Mean Squared Error: 0.07395011580472041
Training with 2 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.15299105819657877
Epoch 200000, Corresponding Error: 0.11012240749618052
Epoch 300000, Corresponding Error: 0.09029740528418936
Epoch 400000, Corresponding Error: 0.08270426452499027
Epoch 500000, Corresponding Error: 0.07552913149383594
Epoch 600000, Corresponding Error: 0.0707927914488291
Epoch 700000, Corresponding Error: 0.0678253990153074
Epoch 800000, Corresponding Error: 0.06586559303360569
Epoch 900000, Corresponding Error: 0.06452316875044938
Epoch 1000000, Corresponding Error: 0.06360186365226636
Model did not converge

Training Mean Squared Error: 0.05066645619729199
Testing Mean Squared Error: 0.05299330984682697
Training with 3 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.15069424995980796
Epoch 200000, Corresponding Error: 0.1212680257099326
```

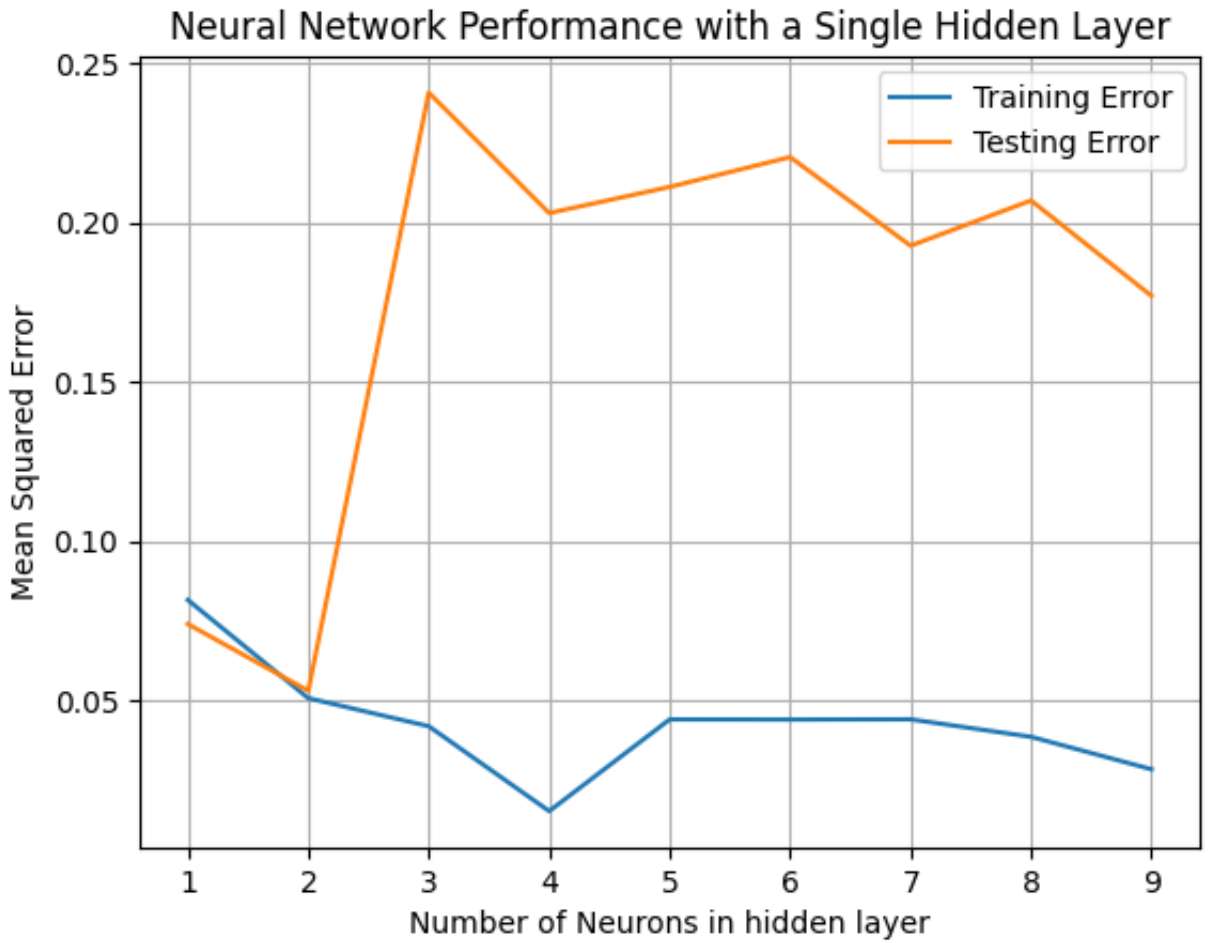
```
Epoch 300000, Corresponding Error: 0.10492658527938584
Epoch 400000, Corresponding Error: 0.09969554438820916
Epoch 500000, Corresponding Error: 0.09613132055607154
Epoch 600000, Corresponding Error: 0.09321036506045824
Epoch 700000, Corresponding Error: 0.0906860884187729
Epoch 800000, Corresponding Error: 0.08849030497784018
Epoch 900000, Corresponding Error: 0.08655881247620982
Epoch 1000000, Corresponding Error: 0.08482526423893041
Model did not converge

Training Mean Squared Error: 0.041862594014615825
Testing Mean Squared Error: 0.24083178464816762
Training with 4 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.16006204631261686
Epoch 200000, Corresponding Error: 0.12996351145563945
Epoch 300000, Corresponding Error: 0.06997487011118261
Epoch 400000, Corresponding Error: 0.05439924048257917
Converged at epoch 462599, Error: 0.04999998079141797

Training Mean Squared Error: 0.015199750418213764
Testing Mean Squared Error: 0.20300208254588936
Training with 5 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.10927026975194587
Epoch 200000, Corresponding Error: 0.08571896257565767
Epoch 300000, Corresponding Error: 0.07261050298176508
Epoch 400000, Corresponding Error: 0.0648215870190648
Epoch 500000, Corresponding Error: 0.06116657986964783
Epoch 600000, Corresponding Error: 0.059107949515026924
Epoch 700000, Corresponding Error: 0.057762005107148184
Epoch 800000, Corresponding Error: 0.05680839844027434
Epoch 900000, Corresponding Error: 0.05609682171116133
Epoch 1000000, Corresponding Error: 0.05554518237682582
Model did not converge

Training Mean Squared Error: 0.043998210523282635
Testing Mean Squared Error: 0.2112069372096595
Training with 6 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.1081281582333637
```

```
Epoch 200000, Corresponding Error: 0.07884170415498522
Epoch 300000, Corresponding Error: 0.06814696807038594
Epoch 400000, Corresponding Error: 0.06287601892772536
Epoch 500000, Corresponding Error: 0.060015742168357235
Epoch 600000, Corresponding Error: 0.05827591303799277
Epoch 700000, Corresponding Error: 0.057115896986434066
Epoch 800000, Corresponding Error: 0.056286969713921275
Epoch 900000, Corresponding Error: 0.05566337029455665
Epoch 1000000, Corresponding Error: 0.05517568731332669
Model did not converge

Training Mean Squared Error: 0.04395557446391099
Testing Mean Squared Error: 0.2205839872115138
Training with 7 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.11164126245792401
Epoch 200000, Corresponding Error: 0.08346892547764927
Epoch 300000, Corresponding Error: 0.06969931324146447
Epoch 400000, Corresponding Error: 0.06425884662910157
Epoch 500000, Corresponding Error: 0.06119592813863517
Epoch 600000, Corresponding Error: 0.05924501321632049
Epoch 700000, Corresponding Error: 0.05791114052176505
Epoch 800000, Corresponding Error: 0.056948724367628004
Epoch 900000, Corresponding Error: 0.05622381443170231
Epoch 1000000, Corresponding Error: 0.05565856058378926
Model did not converge

Training Mean Squared Error: 0.04402499323455501
Testing Mean Squared Error: 0.19275166117929912
Training with 8 neurons in the hidden layer
Epoch 100000, Corresponding Error: 0.10110131772219644
Epoch 300000, Corresponding Error: 0.0641368875875181
Epoch 400000, Corresponding Error: 0.05679700014461696
Epoch 500000, Corresponding Error: 0.05203868592891138
Converged at epoch 565911, Error: 0.049999989247750426

Training Mean Squared Error: 0.03853363818501593
Testing Mean Squared Error: 0.20695081416106093
Training with 9 neurons in the hidden layer
```

```
Epoch 100000, Corresponding Error: 0.1204811366205237
Epoch 200000, Corresponding Error: 0.06631342891115667
Epoch 300000, Corresponding Error: 0.051719535811801956
Converged at epoch 318573, Error: 0.04999997710740796

Training Mean Squared Error: 0.02834824296399847
Testing Mean Squared Error: 0.17700586685492317
```

In [59]:

```
# Plot
plt.plot(range(1, 10), training_errors, label='Training Error')
plt.plot(range(1, 10), testing_errors, label='Testing Error')
plt.xlabel('Number of Neurons in hidden layer')
plt.ylabel('Mean Squared Error')
plt.title('Neural Network Performance with a Single Hidden Layer')
plt.legend()
plt.grid()
plt.show()
```


Neural Network Performance with a Single Hidden Layer
Training Error
Testing Error
Mean Squared Error
0.25
0.20
0.15
0.10
0.05
1
2
3
4
5
6
7
8
9
Number of Neurons in hidden layer

In [ ]: