EXPERIMENT 1

In [14]:
```
import tensorflow as tf
import numpy as np
```

In [15]:
```
# TensorFlow tensor creation
tf_tensor = tf.constant([[1.0, 2.0], [3.0, 4.0]])
print("TensorFlow Tensor:\n", tf_tensor)
```

```
TensorFlow Tensor:
 tf.Tensor(
[[1. 2.]
 [3. 4.]], shape=(2, 2), dtype=float32)
```

In [16]:
```
# NumPy array creation
np_array = np.array([[5.0, 6.0], [7.0, 8.0]])
print("NumPy Array:\n", np_array)
```

```
NumPy Array:
 [[5. 6.]
 [7. 8.]]
```

In [17]:
```
# Convert NumPy to TensorFlow tensor
converted_tensor = tf.convert_to_tensor(np_array)
print("Converted Tensor:\n", converted_tensor)
```

```
Converted Tensor:
 tf.Tensor(
[[5. 6.]
 [7. 8.]], shape=(2, 2), dtype=float64)
```

In [18]:
```
# Mathematical Operations
a = tf.constant([1.0, 2.0, 3.0])
b = tf.constant([4.0, 5.0, 6.0])

print("Add:", tf.add(a, b).numpy())
print("Subtract:", tf.subtract(a, b).numpy())
print("Multiply:", tf.multiply(a, b).numpy())
print("Divide:", tf.divide(a, b).numpy())
print("Power:", tf.pow(a, 2).numpy())
print("Exp:", tf.exp(a).numpy())
print("Sqrt:", tf.sqrt(a).numpy())
```

```
Add: [5. 7. 9.]
Subtract: [-3. -3. -3.]
Multiply: [ 4. 10. 18.]
Divide: [0.25 0.4  0.5 ]
Power: [1. 4. 9.]
Exp: [ 2.7182817  7.389056  20.085537 ]
Sqrt: [1.        1.4142135 1.7320508]
```

In [19]:
```
# Matrix Operations
mat1 = tf.constant([[1, 2], [3, 4]])
mat2 = tf.constant([[5, 6], [7, 8]])

# Matrix multiplication
matmul_result = tf.matmul(mat1, mat2)
print("Matrix Multiplication:\n", matmul_result.numpy())

# Transpose
transpose_result = tf.transpose(mat1)
print("Transpose:\n", transpose_result.numpy())
```

```
Matrix Multiplication:
 [[19 22]
 [43 50]]
Transpose:
 [[1 3]
 [2 4]]
```

In [20]:
```
tensor = tf.constant([[1, 2, 3], [4, 5, 6]])

# Reshape
reshaped = tf.reshape(tensor, [3, 2])
print("Reshaped Tensor:\n", reshaped.numpy())

# Indexing
print("Element at [0, 1]:", tensor[0, 1].numpy())

# Slicing
```

```
print("First row:", tensor[0, :].numpy())
print("Second column:", tensor[:, 1].numpy())

# Reduce sum
reduced_sum = tf.reduce_sum(tensor)
print("Reduced Sum:", reduced_sum.numpy())
```

```
Reshaped Tensor:
 [[1 2]
 [3 4]
 [5 6]]
Element at [0, 1]: 2
First row: [1 2 3]
Second column: [2 5]
Reduced Sum: 21
```

In [21]:

```
# Define a simple function f(x) = x^2
x = tf.Variable(3.0)

with tf.GradientTape() as tape:
    y = x ** 2

# Compute the gradient dy/dx
grad = tape.gradient(y, x)
print("Gradient of x^2 at x=3:", grad.numpy())
```

```
Gradient of x^2 at x=3: 6.0
```