```
print("Assignment 3")
print("Vishnu")
```

```
Assignment 3
Vishnu
```

```
import tensorflow as tf
import numpy as np

m=int(input("Enter the value of m:"))
n=int(input("Enter the value of n:"))
p=int(input("Enter the value of p:"))
```

```
Enter the value of m:3
Enter the value of n:3
Enter the value of p:2
```

```
def create_random_matrix(shape,low,high):
  return tf.random.uniform(shape,minval=low,maxval=high,dtype=tf.float32)

def create_random_matrix_float(shape,low,high):
  return tf.random.uniform(shape,minval=low,maxval=high,dtype=tf.float32)


U = create_random_matrix((m, n), 0, 10)
print("Matrix U:\n", U.numpy())
```

```
Matrix U:
 [[2.55947    5.9210253  4.862052  ]
 [5.7901564  0.38354516 9.665518  ]
 [8.271975   1.5612781  9.995545  ]]
```

```
X = tf.transpose(U)
print("\nMatrix X (transpose of U):\n", X.numpy())
```

```

Matrix X (transpose of U):
 [[2.55947    5.7901564  8.271975  ]
 [5.9210253  0.38354516 1.5612781 ]
 [4.862052   9.665518   9.995545  ]]
```

```
Y = create_random_matrix((1, m), 0, 10)
print("\nMatrix Y:\n", Y.numpy())
```

```

Matrix Y:
 [[2.4841464 8.69969   1.2999475]]
```

```
W1 = create_random_matrix_float((p, n), 0, 1)
print("\nMatrix W1:\n", W1.numpy())
```

```

Matrix W1:
 [[0.8162205  0.8695996  0.08084846]
 [0.22907233 0.05050588 0.26393998]]
```

```
B1 = create_random_matrix_float((p, 1), 0, 1)
print("\nVector B1:\n", B1.numpy())
```

```

Vector B1:
 [[0.2811805 ]
 [0.07548082]]
```

```
W2 = tf.zeros((10, p), dtype=tf.float32)
print("\nMatrix W2:\n", W2.numpy())
```

```

Matrix W2:
 [[0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
```

```
 [0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]]
```

```
B2 = tf.random.uniform([], minval=0, maxval=1, dtype=tf.float32)
print("\nScalar B2:\n", B2.numpy())
```

```
Scalar B2:
 0.9250133
```

```
alpha = 0.01
for i in range(15):

    Z1 = tf.matmul(W1, X) + B1

    A1 = tf.nn.relu(Z1)

    Z2 = tf.matmul(W2, A1) + B2

    A2 = tf.nn.softmax(Z2, axis=0)

    # Cast Y to integers before using tf.one_hot
    Y_one_hot = tf.one_hot(tf.cast(tf.squeeze(Y), dtype=tf.int32), depth=10)
    Y_one_hot = tf.transpose(Y_one_hot)
    dZ2 = A2 - Y_one_hot

    dA2 = tf.matmul(W2, dZ2, transpose_a=True)

    dW2 = tf.matmul(dZ2, A1, transpose_b=True) / m

    dB2 = tf.reduce_mean(dZ2, axis=1, keepdims=True)

    dZ1 = dA2 * tf.cast(Z1 > 0, dtype=tf.float32)

    dA1 = tf.matmul(W1, dZ1, transpose_a=True)

    dB1 = tf.reduce_mean(dZ1, axis=1, keepdims=True)

    dW1 = tf.matmul(dZ1, X, transpose_b=True) / m

    W1 = W1 - alpha * dW1
    B1 = B1 - alpha * dB1
    W2 = W2 - alpha * dW2
    B2 = B2 - alpha * dB2

    print(f"\nIteration {i+1}:\nW1:\n{W1.numpy()}\nB1:\n{B1.numpy()}\nW2:\n{W2.numpy()}\nB2:\n{B2.numpy()}")
```

```
Iteration 1:
W1:
[[0.8162205  0.8695996  0.08084846]
 [0.22907233 0.05050588 0.26393998]]
B1:
[[0.2811805 ]
 [0.07548082]]
W2:
[[-0.00774441 -0.00363463]
 [ 0.02291808  0.01199017]
 [ 0.01862987  0.00384577]
 [-0.00774441 -0.00363463]
 [-0.00774441 -0.00363463]
 [-0.00774441 -0.00363463]
 [-0.00774441 -0.00363463]
 [-0.00774441 -0.00363463]
 [ 0.01266292  0.00960648]
 [-0.00774441 -0.00363463]]
B2:
[[0.9240133 ]
 [0.92734665]
 [0.92734665]
 [0.9240133 ]
 [0.9240133 ]
 [0.9240133 ]
 [0.9240133 ]
 [0.9240133 ]
 [0.92734665]
```

```
 [0.9240133 ]]

Iteration 2:
W1:
[[0.81717145 0.8700646  0.08220098]
 [0.22958158 0.0506387  0.26465455]]
B1:
[[0.2813466 ]
 [0.07555878]]
W2:
[[-0.01487078 -0.00697966]
 [ 0.04391459  0.02307842]
 [ 0.03593938  0.00707827]
 [-0.01487078 -0.00697966]
 [-0.01487078 -0.00697966]
 [-0.01487078 -0.00697966]
 [-0.01487078 -0.00697966]
 [-0.01487078 -0.00697966]
 [ 0.02424151  0.01870095]
 [-0.01487078 -0.00697966]]
B2:
[[0.92309123]
 [0.9294378 ]
 [0.9295137 ]
 [0.92309123]
 [0.92309123]
 [0.92309123]
 [0.92309123]
 [0.92309123]
```