```
def hebbian_learning(samples):
     print(f'{"INPUT":^8} {"TARGET":^16}{"WEIGHT CHANGES":^15}{"WEIGHTS":^25}')
     w1, w2, b = 0, 0, 0
     print(' ' * 45, f'({w1:2}, {w2:2}, {b:2})')
     for x1, x2, y in samples:
         w1 = w1 + x1 * y
         w2 = w2 + x2 * y
         b = b + y
         print(f'({x1:2}, {x2:2})       {y:2}         ({x1*y:2}, {x2*y:2}, {y:2})        ({w1:2}, {w2:2}, {b:2})')


AND_samples = {
    'binary_input_binary_output': [
        [1, 1, 1],
        [1, 0, 0],
        [0, 1, 0],
        [0, 0, 0]
    ],
    'binary_input_bipolar_output': [
        [1, 1, 1],
        [1, 0, -1],
        [0, 1, -1],
        [0, 0, -1]
    ],
    'bipolar_input_bipolar_output': [
        [ 1, 1, 1],
        [ 1, -1, -1],
        [-1, 1, -1],
        [-1, -1, -1]
    ]
}

OR_samples = {
    'binary_input_binary_output': [
        [1, 1, 1],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, 0]
    ],
    'binary_input_bipolar_output': [
        [1, 1, 1],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, -1]
    ],
    'bipolar_input_bipolar_output': [
        [ 1, 1, 1],
        [ 1, -1, 1],
        [-1, 1, 1],
        [-1, -1, -1]
    ]
}
XOR_samples = {
    'binary_input_binary_output': [
        [1, 1, 0],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, 0]
    ],
    'binary_input_bipolar_output': [
        [1, 1, -1],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, -1]
    ],
    'bipolar_input_bipolar_output': [
        [ 1, 1, -1],
        [ 1, -1, 1],
        [-1, 1, 1],
        [-1, -1, -1]
    ]
}


print('AND with Binary Input and Binary Output\n')
hebbian_learning(AND_samples['binary_input_binary_output'])
```

```
print('\nAND with Binary Input and Bipolar Output\n')
hebbian_learning(AND_samples['binary_input_bipolar_output'])

print('\nAND with Bipolar Input and Bipolar Output\n')
hebbian_learning(AND_samples['bipolar_input_bipolar_output'])
```

```
     AND with Binary Input and Binary Output

      INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
     ( 1,  1)        1         ( 1,  1,  1)       ( 1,  1,  1)
     ( 1,  0)        0         ( 0,  0,  0)       ( 1,  1,  1)
     ( 0,  1)        0         ( 0,  0,  0)       ( 1,  1,  1)
     ( 0,  0)        0         ( 0,  0,  0)       ( 1,  1,  1)

     AND with Binary Input and Bipolar Output

      INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
     ( 1,  1)        1         ( 1,  1,  1)       ( 1,  1,  1)
     ( 1,  0)       -1         (-1,  0, -1)       ( 0,  1,  0)
     ( 0,  1)       -1         ( 0, -1, -1)       ( 0,  0, -1)
     ( 0,  0)       -1         ( 0,  0, -1)       ( 0,  0, -2)

     AND with Bipolar Input and Bipolar Output

      INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
     ( 1,  1)        1         ( 1,  1,  1)       ( 1,  1,  1)
     ( 1, -1)       -1         (-1,  1, -1)       ( 0,  2,  0)
     (-1,  1)       -1         ( 1, -1, -1)       ( 1,  1, -1)
     (-1, -1)       -1         ( 1,  1, -1)       ( 2,  2, -2)
```

```
print('OR with binary input and binary output\n')
hebbian_learning(OR_samples['binary_input_binary_output'])

print('\nOR with binary input and bipolar output\n')
hebbian_learning(OR_samples['binary_input_bipolar_output'])

print('\nOR with bipolar input and bipolar output\n')
hebbian_learning(OR_samples['bipolar_input_bipolar_output'])
```

```
     OR with binary input and binary output

      INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
     ( 1,  1)        1         ( 1,  1,  1)       ( 1,  1,  1)
     ( 1,  0)        1         ( 1,  0,  1)       ( 2,  1,  2)
     ( 0,  1)        1         ( 0,  1,  1)       ( 2,  2,  3)
     ( 0,  0)        0         ( 0,  0,  0)       ( 2,  2,  3)

     OR with binary input and bipolar output

      INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
     ( 1,  1)        1         ( 1,  1,  1)       ( 1,  1,  1)
     ( 1,  0)        1         ( 1,  0,  1)       ( 2,  1,  2)
     ( 0,  1)        1         ( 0,  1,  1)       ( 2,  2,  3)
     ( 0,  0)       -1         ( 0,  0, -1)       ( 2,  2,  2)

     OR with bipolar input and bipolar output

      INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
     ( 1,  1)        1         ( 1,  1,  1)       ( 1,  1,  1)
     ( 1, -1)        1         ( 1, -1,  1)       ( 2,  0,  2)
     (-1,  1)        1         (-1,  1,  1)       ( 1,  1,  3)
     (-1, -1)       -1         ( 1,  1, -1)       ( 2,  2,  2)
```

```
print('XOR with binary input and binary output\n')
hebbian_learning(XOR_samples['binary_input_binary_output'])

print('\nXOR with binary input and bipolar output\n')
hebbian_learning(XOR_samples['binary_input_bipolar_output'])
```

```
print('\nXOR with bipolar input and bipolar output\n')
hebbian_learning(XOR_samples['bipolar_input_bipolar_output'])
```

```
    XOR with binary input and binary output

     INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
    ( 1,  1)        0         ( 0,  0,  0)       ( 0,  0,  0)
    ( 1,  0)        1         ( 1,  0,  1)       ( 1,  0,  1)
    ( 0,  1)        1         ( 0,  1,  1)       ( 1,  1,  2)
    ( 0,  0)        0         ( 0,  0,  0)       ( 1,  1,  2)

    XOR with binary input and bipolar output

     INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
    ( 1,  1)       -1         (-1, -1, -1)       (-1, -1, -1)
    ( 1,  0)        1         ( 1,  0,  1)       ( 0, -1,  0)
    ( 0,  1)        1         ( 0,  1,  1)       ( 0,  0,  1)
    ( 0,  0)       -1         ( 0,  0, -1)       ( 0,  0,  0)

    XOR with bipolar input and bipolar output

     INPUT        TARGET     WEIGHT CHANGES          WEIGHTS
                                                  ( 0,  0,  0)
    ( 1,  1)       -1         (-1, -1, -1)       (-1, -1, -1)
    ( 1, -1)        1         ( 1, -1,  1)       ( 0, -2,  0)
    (-1,  1)        1         (-1,  1,  1)       (-1, -1,  1)
    (-1, -1)       -1         ( 1,  1, -1)       ( 0,  0,  0)
```

✓ 0s completed at 6:01 AM