Q21. Project XOR design

Step 1: Study the general idea on how to design XOR Gate

OR				NAND				XOR		
X	Y	Z1		X	Y	Z2		X	Y	Z3
0	0	0		0	0	1		0	0	0
0	1	1	AND	0	1	1	=	0	1	1
1	0	1		1	0	1		1	0	1
1	1	1		1	1	0		1	1	0

Our neural network equation can be created by combining neural equations.

```
Z1 := X  "Or"   Y
Z2 := X  "NAND" Y
Z := Z3 := Z1 "AND" Z2
Z := ( X "Or" Y ) "AND" ( X "NAND" Y )
```

Step 2: Using the following rules to design your own AND Gate, OR Gate, and NAND Gate

a) Train OR gate to get W1, W2, Y

Desired
Function

OR

X	Y	Z1
0	0	0
0	1	1
1	0	1
1	1	1

Train OR gate to get W1, W2, Y

```
Z1:= ( W1 * X + W2 * Y >= T )
   where T := 1.0.
    Loop 1
    W1=W2=0
    Function
    X Y | Z1
   ----------
    0 0 | 0

    0 1 | 0

    1 0 | 0

    1 1 | 0

    Loop 2
    W1=W2=0.5
    Function
    X Y | Z1
   ----------
    0 0 | 0

    0 1 | 0

    1 0 | 0

    1 1 | 1
```

Loop 3
W1=W2=1.0
Function

X	Y	Z1
0	0	0
0	1	1
1	0	1
1	1	1

Z 1:= (1 * X + 1 * Y >= 1.0)

B) Train NAND gate to get W1, W2, Y

Desired

Function

NAND

X	Y	Z2
0	0	1
0	1	1
1	0	1
1	1	0

Loop 1
W0=0
W1=W2=0.5
Function

C	X	Y	Z2
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

*As We got 0 for z to get desired output we are using forward process for W0, W1 and W2

Loop 2
W0=0.5
W1=W2=0.5
Function

C	X	Y	Z2
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Loop 3
W0=1
W1=W2=0.5
Function

```
C X Y | Z2
----------
1 0 0 | 1

1 0 1 | 1

1 1 0 | 1

1 1 1 | 1
```

*As We got 1 for z we are using backward process for W1 and W2

Loop 4
W0=1
W1=W2=0.0
Function

```
C X Y | Z2

----------
1 0 0 | 1

1 0 1 | 1

1 1 0 | 1

1 1 1 | 1
```

*As We got 1 for z we are using backward process for W1 and W2

Loop 5

W0=1

W1=W2=-0.5

Function

C	X	Y	Z2
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

*As We got 0 for z we are using forward process for W0 and backward process for W1 and W2

Loop 6

W0=1.5

W1=W2=-0.5

Function

C	X	Y	Z2
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

NAND Formula
C for NAND is 1.0
==> Z2 := 1.5 * 1 + (-0.5) * X + (-0.5)* Y >= 1

c) Train AND gate to get W1, W2, Y
Desired

Function
AND

X	Y	Z3
0	0	0
0	1	0
1	0	0
1	1	1

Train AND gate to get W1, W2, Y

```
Z1:= ( W1 * X + W2 * Y >= T )
   where T := 1.0.
    Loop 1
    W1=W2=0
    Function
```

X	Y	Z3
0	0	0
0	1	0
1	0	0

```
 1 1 | 0

 Loop 2
 W1=W2=0.5
 Function
 X Y | Z3
----------
 0 0 | 0

 0 1 | 0

 1 0 | 0

 1 1 | 1
```

Z3 := (0.5 * X + 0.5 * Y >= 1.0)

Step 3: Please answer

Formula for OR Z1 := X "OR" Y

Z1:= (1 * X + 1 * Y >= 1.0)

Formula for NAND Z2 := X "OR" Y

Z2:= 1.5 * 1 + (-0.5) * X + (-0.5)* Y >= 1

Formula for AND Z3 := X "AND" Y

Z3 := (0.5 * X + 0.5 * Y >= 1.0)

Formula for XOR Z := Z3 := Z1 "AND" Z2

```
Z1 := X  "Or"   Y
Z2 := X  "NAND" Y
Z := Z3 := Z1 "AND" Z2
Z := ( X "Or" Y ) "AND" ( X "NAND" Y )
Z := ( 0.5 * ( 1.0 * X + 1.0 * Y >= 1.0 ) "AND"
0.5 * ( 1.5*1 + -0.5 * X + -0.5 * Y >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 * X + 1.0 * Y >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * X + -0.5 * Y >= 1.0) >= 1.0 )
```

Step 4: Please prove that your designed XOR Gate work

a) X=1, Y=1

```
Z := ( 0.5 * ( 1.0 * X + 1.0 * Y >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * X + -0.5 * Y >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 * 1 + 1.0 * 1 >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * 1 + -0.5 * 1 >= 1.0) >= 1.0 )

Z := ( 0.5 * ( 1.0  + 1.0 >= 1.0 ) + 0.5 * ( 1.5+ -0.5 + -0.5 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 2.0 >= 1.0 ) + 0.5 * ( 0.5 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( true ) + 0.5 * ( false) >= 1.0 )
Z := ( 0.5 * 1.0 + 0.5 * 0.0 >= 1.0 )
Z := ( 0.5 >= 1.0 )
Z := ( false )
Z := 0
```

b)X=1, Y=0

```
Z := ( 0.5 * ( 1.0 * X + 1.0 * Y >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * X + -0.5 * Y >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 * 1 + 1.0 * 0 >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * 1 + -0.5 * 0 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 + 0.0 >= 1.0 ) + 0.5 * ( 1.5+ -0.5 >= 1.0) >= 1.0 )
```

```
Z := ( 0.5 * ( 1.0 >= 1.0 ) + 0.5 * ( 1.0 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( true ) + 0.5 * (true) >= 1.0 )
Z := ( 0.5 * 1.0 + 0.5 * 1.0 >= 1.0 )
Z := ( 1.0 >= 1.0 )
Z := ( true )
Z := 1
```

c) X=0, Y=1

```
Z := ( 0.5 * ( 1.0 * X + 1.0 * Y >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * X + -0.5 * Y >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 * 0 + 1.0 * 1 >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * 0 + -0.5 * 1 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 + 0.0 >= 1.0 ) + 0.5 * ( 1.5+ -0.5 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 >= 1.0 ) + 0.5 * ( 1.0 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( true ) + 0.5 * (true) >= 1.0 )
Z := ( 0.5 * 1.0 + 0.5 * 1.0 >= 1.0 )
Z := ( 1.0 >= 1.0 )
Z := ( true )
Z := 1
```

d) X=0, Y=0

```
Z := ( 0.5 * ( 1.0 * X + 1.0 * Y >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * X + -0.5 * Y >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 1.0 * 0 + 1.0 * 0 >= 1.0 ) +
0.5 * ( 1.5*1 + -0.5 * 0 + -0.5 * 0 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 0.0  + 0.0 >= 1.0 ) + 0.5 * ( 1.5+ -0.0 + -0.0 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( 0.0 >= 1.0 ) + 0.5 * ( 1.5 >= 1.0) >= 1.0 )
Z := ( 0.5 * ( false ) + 0.5 * ( true ) >= 1.0 )
Z := ( 0.5 * 0.0 + 0.5 * 1.0 >= 1.0 )
Z := ( 0.5 >= 1.0 )
Z := ( false )
Z := 0
```

OR

X	Y	Z1
0	0	0
0	1	1
1	0	1
1	1	1

AND

NAND

X	Y	Z2
0	0	1
0	1	1
1	0	1
1	1	0

=

XOR

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

Our black box for the "XOR" function now has three neurons in it. A collection of neurons connected together is a "network" of neurons. Thus, the "XOR" function has been created using a "neural network".

```
      +------------------------------------------------------------+
      |      "NAND"  +--------+                    "XOR" Function  |
      | 1.0-->1.5--->|        |                                    |
      |              |        |                                    |
X --->|----> -0.5-->| Neuron  |-->Z2--                             |
      |\             |        |       \                            |
Y---->|-\--> -0.5-->|         |        \ "AND"  +--------+         |
      |\ \           +--------+         ->0.5-->|        |         |
      | \ \                                     | Neuron |---->Z3->| -->Z
      |  \ \  "Or"   +--------+        -->0.5-->|        |         |
      |   \ -->1.0-->|        |       /         +--------+         |
      |    \         | Neuron |-->Z1--                             |
      |     -->1.0-->|        |                                    |
      |              +--------+                                    |
      +------------------------------------------------------------+
```

* Hence proved that the neural network works for XOR Gate