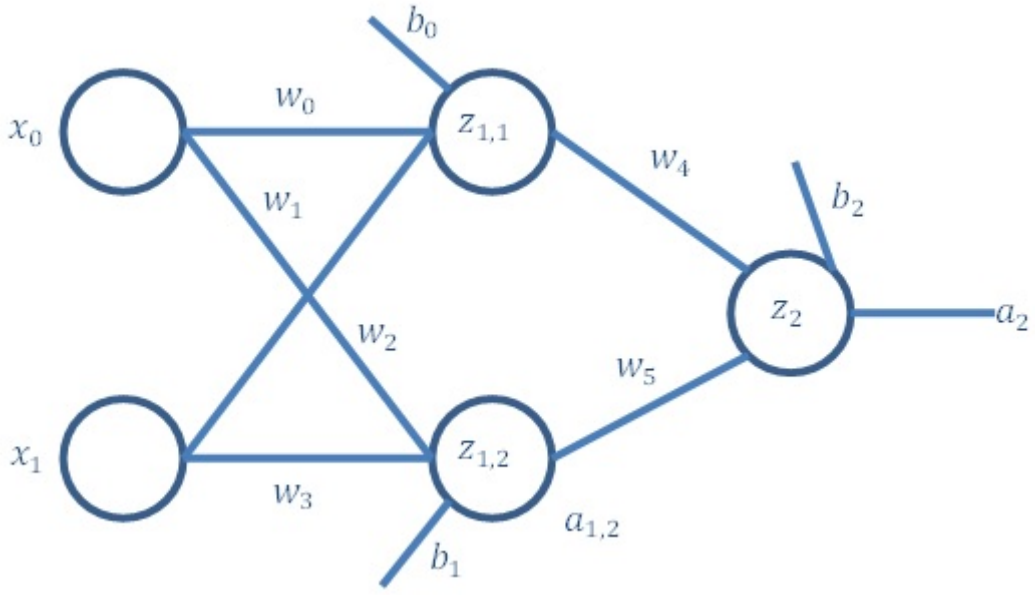
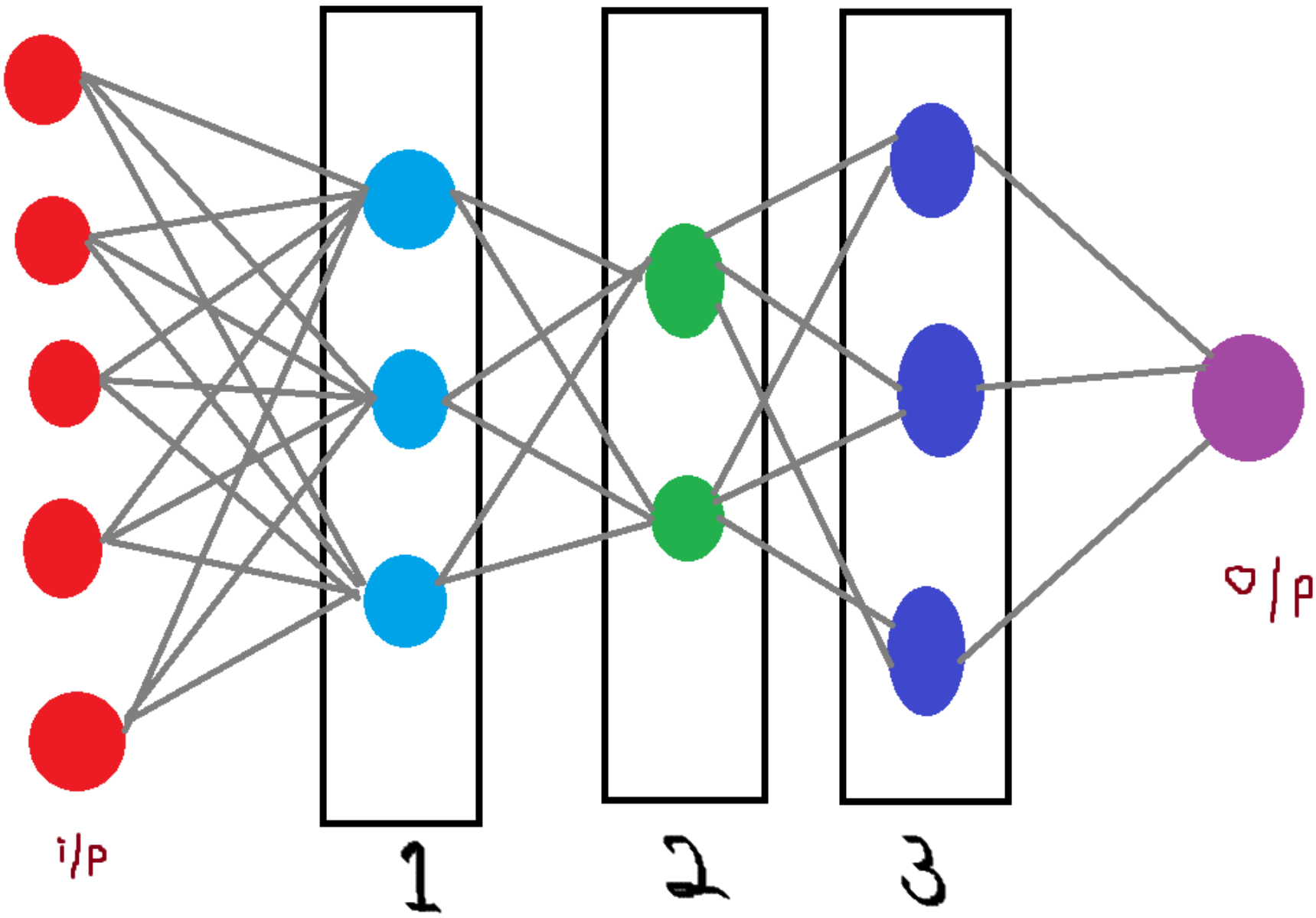


In [65]:
```
# week 1
import numpy as np
n=2;h=1;m=2;o=1
w1=np.around(np.random.uniform(size=n*m),decimals=2)
b1=np.around(np.random.uniform(size=m),decimals=2)
w2=np.around(np.random.uniform(size=m*o),decimals=2)
b2=np.around(np.random.uniform(size=o),decimals=2)
print("Weights between Input & Hidden layer:",w1)
print("Bias in Hidden layer:",b1)
print("Weights between Hidden & Output layer:",w2)
print("Bias in Output layer:",b2)
```

```
Weights between Input & Hidden layer: [0.61 0.6  0.55 0.93]
Bias in Hidden layer: [0.51 0.32]
Weights between Hidden & Output layer: [0.41 0.24]
Bias in Output layer: [0.73]
```

In [108]:
```
# week 2
import numpy as np
n=2;h=1;m=2;o=1
w=np.around(np.random.uniform(size=6),decimals=2)
b=np.around(np.random.uniform(size=3),decimals=2)
print("weights",w)
print("biases",b)
x0=0.5;x1=0.85
print("x0 is",x_0,"and x1 is",x_1)
z11=x0*w[0]+x1*w[1]+b[0]
z12=x0*w[2]+x1*w[3]+b[1]
print("The weighted sum of the inputs at the first node in the hidden layer is",np.around(z11,decimals=2))
print("The weighted sum of the inputs at the second node in the hidden layer is",np.around(z12,decimals=2))
a11=1.0/(1.0+np.exp(-z11))
a12=1.0/(1.0+np.exp(-z12))
print("The activation of the first node in the hidden layer is",np.around(a11,decimals=2))
print("The activation of the second node in the hidden layer is",np.around(a12,decimals=2))
z2=a11*w[4]+a12*w[5]+b[2]
print("The weighted sum of the inputs at the node in the output layer is",np.around(z2,decimals=2))
a2=1.0/(1.0+np.exp(-z2))
print("The output of the network for x0(0.5) and x1(0.85) is",np.around(a2,decimals=2))
```

```
weights [0.96 0.26 0.24 0.69 0.72 0.02]
biases [0.43 0.66 0.12]
x0 is 0.5 and x1 is 0.85
The weighted sum of the inputs at the first node in the hidden layer is 1.13
The weighted sum of the inputs at the second node in the hidden layer is 1.37
The activation of the first node in the hidden layer is 0.76
The activation of the second node in the hidden layer is 0.8
The weighted sum of the inputs at the node in the output layer is 0.68
The output of the network for x0(0.5) and x1(0.85) is 0.66
```



In [144]:
```
# week 3
import numpy as np
n=5;h=3;m=[3,2,3];o=1
ip=np.around(np.random.uniform(size=n),decimals=2)
def create_network(n,h,m,o):
    pn=n
    res={}
    for i in range(h+1):
        if i==h:
            ln="Output"
            nn=o
        else:
            ln="Layer-"+str(i+1)
            nn=m[i]
        res[ln]={}
        for j in range(nn):
            ss="Node-"+str(j+1)
            res[ln][ss]={
                'Weights':np.around(np.random.uniform(size=pn),decimals=2),
                'Bias':np.around(np.random.uniform(size=1),decimals=2),
                }
            pn=nn
    return res
network=create_network(n,h,m,o)
print(network)
```

```
{'Layer-1': {'Node-1': {'Weights': array([0.9 , 0.5 , 0.23, 0.68, 0.03]), 'Bias': array([0.92])}, 'Node-2': {'Weight
s': array([0.57, 0.9 , 0.64, 0.67, 0.79]), 'Bias': array([0.18])}, 'Node-3': {'Weights': array([0.56, 0.03, 0.18, 0.
28, 0.1 ]), 'Bias': array([0.02])}}, 'Layer-2': {'Node-1': {'Weights': array([0.17, 0.42, 0.68]), 'Bias': array([0.8
7])}, 'Node-2': {'Weights': array([0.63, 0.73, 0.47]), 'Bias': array([0.28])}}, 'Layer-3': {'Node-1': {'Weights': ar
ray([0.44, 0.54]), 'Bias': array([0.97])}, 'Node-2': {'Weights': array([0.93, 0.69]), 'Bias': array([0.5])}, 'Node-
3': {'Weights': array([0.03, 0.73]), 'Bias': array([0.43])}}, 'Output': {'Node-1': {'Weights': array([0.65, 0.08, 0.
84]), 'Bias': array([0.81])}}}
```

In [172]:
```
# week 4-a
import numpy as np
n=5;h=3;m=[3,2,3];o=1
ip=np.around(np.random.uniform(size=n),decimals=2)
def create_network(n,h,m,o):
    pn=n
    res={}
    for i in range(h+1):
        if i==h:
            ln="Output"
            nn=o
        else:
            ln="Layer-"+str(i+1)
            nn=m[i]
        res[ln]={}
        for j in range(nn):
            ss="Node-"+str(j+1)
            res[ln][ss]={
                'Weights':np.around(np.random.uniform(size=pn),decimals=2),
                'Bias':np.around(np.random.uniform(size=1),decimals=2),
                }
        pn=nn
    return res
def CWS(ip,weights,bias):
    return np.sum(ip*weights)+bias
def NA(ws):
    return 1.0/(1.0+np.exp(-ws))
print("The inputs to the network are",ip)
def FP(network,ip):
    li=list(ip)
    for i in network:
        ld=network[i]
        lo=[]
        for j in ld:
            nd=ld[j]
            no=NA(CWS(li,nd['Weights'],nd['Bias']))
            lo.append(np.around(no[0],decimals=2))
        if i!="Output":
             print("The outputs of the nodes in hidden layer",i.split('-')[1],"is",lo)
        li=lo
    return lo
network=create_network(n,h,m,o)
pred=FP(network,ip)
print("The predicted value by the network for the given input is",np.around(pred,decimals=2))
```

```
The inputs to the network are [0.21 0.87 0.21 0.2  0.71]
The outputs of the nodes in hidden layer 1 is [0.84, 0.84, 0.73]
The outputs of the nodes in hidden layer 2 is [0.65, 0.9]
The outputs of the nodes in hidden layer 3 is [0.72, 0.88, 0.81]
The predicted value by the network for the given input is [0.83]
```

In [153]:
```
# week 4-b
import numpy as np
n=5;h=3;m=[3,2,3];o=1
ip=np.around(np.random.uniform(size=n),decimals=2)
def create_network(n,h,m,o):
    pn=n
    res={}
    for i in range(h+1):
        if i==h:
            ln="Output"
            nn=o
        else:
            ln="Layer-"+str(i+1)
            nn=m[i]
        res[ln]={}
        for j in range(nn):
            ss="Node-"+str(j+1)
            res[ln][ss]={
                'Weights':np.around(np.random.uniform(size=pn),decimals=2),
                'Bias':np.around(np.random.uniform(size=1),decimals=2),
                }
        pn=nn
    return res
def CWS(ip,weights,bias):
    return np.sum(ip*weights)+bias
def NA(ws):
    return (np.exp(ws)-np.exp(-ws))/(np.exp(ws)+np.exp(-ws))
print("The inputs to the network are",ip)
def FP(network,ip):
    li=list(ip)
    for i in network:
        ld=network[i]
        lo=[]
        for j in ld:
            nd=ld[j]
            no=NA(CWS(li,nd['Weights'],nd['Bias']))
            lo.append(np.around(no[0],decimals=2))
        if i!="Output":
             print("The outputs of the nodes in hidden layer",i.split('-')[1],"is",lo)
        li=lo
    return lo
network=create_network(n,h,m,o)
pred=FP(network,ip)
print("The predicted value by the network for the given input is",np.around(pred,decimals=2))
```

```
The inputs to the network are [0.77 0.17 0.79 0.15 0.47]
The outputs of the nodes in hidden layer 1 is [0.96, 0.88, 0.89]
The outputs of the nodes in hidden layer 2 is [0.97, 0.99]
The outputs of the nodes in hidden layer 3 is [0.96, 0.48, 0.66]
The predicted value by the network for the given input is [0.94]
```

In [183]:
```
# week 5
import numpy as np
n=5;h=3;m=[3,2,3];o=1
ip=np.around(np.random.uniform(size=n),decimals=2)
def create_network(n,h,m,o):
    pn=n
    res={}
    for i in range(h+1):
        if i==h:
            ln="Output"
            nn=o
        else:
            ln="Layer-"+str(i+1)
            nn=m[i]
        res[ln]={}
        for j in range(nn):
            ss="Node-"+str(j+1)
            res[ln][ss]={
                'Weights':np.around(np.random.uniform(size=pn),decimals=2),
                'Bias':np.around(np.random.uniform(size=1),decimals=2),
                }
        pn=nn
    return res
def CWS(ip,weights,bias):
    return np.sum(ip*weights)+bias
def NA(ws):
    return np.maximum(0,ws)
print("The inputs to the network are",ip)
def FP(network,ip):
    li=list(ip)
    for i in network:
        ld=network[i]
        lo=[]
        for j in ld:
            nd=ld[j]
            no=NA(CWS(li,nd['Weights'],nd['Bias']))
            lo.append(np.around(no[0],decimals=2))
        if i!="Output":
             print("The outputs of the nodes in hidden layer",i.split('-')[1],"is",lo)
        li=lo
    return lo
network=create_network(n,h,m,o)
pred=FP(network,ip)
print("The predicted value by the network for the given input is",np.around(pred,decimals=2))
```

```
The inputs to the network are [0.1  0.06 0.72 0.43 0.64]
The outputs of the nodes in hidden layer 1 is [1.68, 1.69, 1.64]
The outputs of the nodes in hidden layer 2 is [2.67, 3.68]
The outputs of the nodes in hidden layer 3 is [5.9, 4.06, 2.19]
The predicted value by the network for the given input is [7.25]
```