Assignment 8

In [1]:
```
import numpy as np
```

In [2]:
```
#define the ART network class
class ARTNetwork:

    def __init__(self, input_size, rho, alpha):
        self.input_size = input_size
        self.rho = rho
        self.alpha = alpha
        self.W = np.zeros(input_size)
        self.V = np.ones(input_size)

    def train(self, X):
        for x in X:

            #calculate the activation
            y = x / (self.rho + np.linalg.norm(self.W))

            # find the index of the maximum activation
            j = np.argmax(y)

            # if the maximum activation is greater than or equal to alpha times
            # the sum of all activations and the vigilance parameter at that
            # index is greater than 0
            if y[j] >= self.alpha * np.sum(y) and self.V[j] > 0:

                 #update the weight vector
                self.W += self.V[j] * x
                self.V[j] *= 0.5

            else:
                self.V[j] += 0.5

    def classify(self, X):
        classes = []
        for x in X:

            #calculate the activation
            y = x / (self.rho + np.linalg.norm(self.W))

            #find the index of the maximum activation and append it to classes
            j = np.argmax(y)
            classes.append(j)

        return classes
```

In [3]:

```
# create some sample input data
X_train = np.array([[0, 1, 1, 0],
                    [1, 0, 0, 1],
                    [1, 0, 0, 0]])

X_test = np.array([[0, 1, 0, 0],
                   [1, 1, 1, 0]])

# initialize the ARTNetwork with input size, rho, and alpha
input_size = X_train.shape[1]
rho = 0.5
alpha = 0.9
art_network = ARTNetwork(input_size, rho, alpha)

# train the network on the input data
art_network.train(X_train)

# classify the test data using the learned network
classes = art_network.classify(X_test)

# print out the predicted classes for each data point in X_test
for i, data in enumerate(X_test):
    print(f"Test Data {i+1}: Predicted Class: {classes[i]}")
```

```
Test Data 1: Predicted Class: 1
Test Data 2: Predicted Class: 0
```