1. Initialisation des variables :

- w1 et w2 : Les poids synaptiques associés aux entrées x1 et x2, initialisés aléatoirement entre 0 et 1.
- n : Le taux d'apprentissage, fixé à 0.7.
- L1 et L2 : Les entrées possibles pour le perceptron (les combinaisons binaires de la table de vérité OR).

2. Fonctions définies :

- f(x) : Fonction d'activation qui renvoie 1 si x >= 0 et 0 sinon. C'est une fonction de seuil utilisée pour prendre des décisions.
- OR(x1, x2) : Calcule la sortie en utilisant les poids actuels et applique la fonction de seuil. La sortie est 1 si la somme pondérée des entrées moins le biais est positive, sinon 0.
- ran(x, y) : Met à jour les poids w1 et w2 si la sortie calculée est différente de la sortie attendue. L'apprentissage se fait en ajustant les poids selon la règle de mise à jour suivante :

$$w_{nouveau} = w_{ancien} + n \times ((valeur\_attendue) - (sortie\_obtenue)) \times (entrée)$$

3. Boucle d'apprentissage :

- La boucle while True permet de continuer l'entraînement jusqu'à ce que le perceptron trouve les poids corrects pour modéliser la fonction OR sans erreur.
- La variable erreur est incrémentée si une erreur est détectée (c'est-à-dire si la sortie calculée ne correspond pas à la sortie attendue).
- Si erreur == 0 à la fin d'une itération complète de la table de vérité, le modèle a trouvé les poids corrects et l'entraînement s'arrête.

4. Sortie des résultats :

Le programme imprime les poids finaux trouvés et affiche les résultats de la fonction OR pour chaque combinaison de x1 et x2.

Résumé :

Ce code simule un **perceptron** simple capable d'apprendre à modéliser la fonction **OR** logique à travers un processus itératif d'entraînement. Il ajuste les poids w1 et w2 jusqu'à ce que le perceptron soit capable de produire la sortie correcte pour toutes les combinaisons possibles d'entrées.