**PC.2. Implemente, utilizando a linguagem de programação Python ou JavaScript, uma**

**rede neural que resolva o problema do AND lógico. Seu programa não precisa**

**efetuar nenhum processo de aprendizagem e pode utilizar pesos definidos de modo**

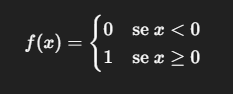
**arbitrário por você. Ilustre dois casos utilizando as seguintes funções de ativação:**

**função de Heaviside e a função ReLU. Diga se os pesos utilizados foram os mesmos**

**quando houve alteração da função de ativação.**

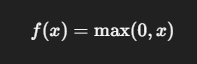
### **O que é uma função de ativação?**

É uma função matemática usada por um neurônio para decidir **se ativa (dispara)** ou não, dado um valor de entrada.

**Função de Ativação Heaviside (Degrau):**  
Também chamada de **função degrau**, ativa o neurônio somente quando a entrada atinge ou ultrapassa um limite, ideal para: lógica booleana.  
Fórmula:   


**Função de Ativação ReLU (Rectified Linear Unit):**

A função ReLU só deixa passar valores positivos. Tudo que é menor que zero vira zero, ideal para: deep learning e regressão.

Fórmula:  


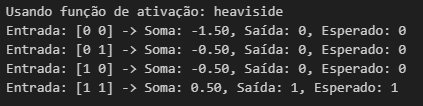


**mesmos pesos arbitrários** [1, 1, -1.5] em ambos os casos.



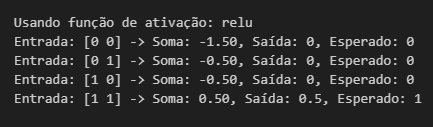
Função de ativação Heaviside:

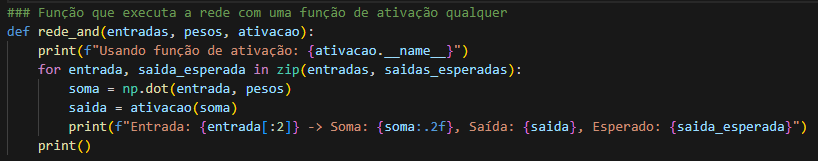




Função de ativação ReLu:







A diferença nos resultados vêm da **função de ativação**:

* A **Heaviside** produz **valores binários** (0 ou 1), ideal para lógica booleana como o AND.
* A **ReLU** produz **valores contínuos ≥ 0**, o que pode não coincidir com as saídas esperadas sem um limiar adicional.

**O que mudou foi apenas a função de ativação**, e não os pesos. Isso demonstra que a saída final da rede depende não só dos pesos e entradas, mas também da função de ativação aplicada.