Algorithm: VDN Training

Initialize $\theta$, the parameters of individual Q-networks, and $\theta^-$ the parameters of target networks.
Initialize replay buffer $\mathcal{D}$ // $(\mathbf{o}_t, \mathbf{a}_t, r_t, done, \mathbf{o}_{t+1})$
**while** $t < T$ **do**
    Collect observations $\{o_1^t, \ldots, o_n^t\}$
    **for** each agent $i$ **do**
        With probability $\epsilon$, select random action $a_i^t$
        otherwise select $a_i^t = \arg\max_{a_i} Q_i(o_i^t, a_i)$
    **end for**
    Execute joint action $\mathbf{a}^t = (a_1^t, \ldots, a_n^t)$
    Collect $r^t$, $done^t$, and $\mathbf{o}^{t+1}$
    Store $(\mathbf{o}^t, \mathbf{a}^t, r^t, done^t, \mathbf{o}^{t+1})$ in $\mathcal{D}$
    **if** $t$ is a training step **then**
        Sample batch $\mathcal{B} = \{\mathbf{o}^b, \mathbf{a}^b, r^b, done^b, \mathbf{o'}^b\}$
        Set the targets

$$y^b = r^b + \gamma(1 - done^b) \sum_i \max_{a_i'} Q_i(o_i^b, a_i'; \theta^-)$$

        Perform a gradient descent using:

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|} \sum_b \left( y^b - \sum_i Q_i(o_i^b, a_i^b; \theta) \right)^2$$

        Every $C$ steps, update $\theta^- \leftarrow \theta$
    **end if**
**end while**