Algorithm 1: Randomized Ensembled Double Q-Learning (REDQ)

Input: Initial policy parameters θ, ensemble of M Q-functions $\{\phi_i\}_{i=1}^{M}$, target Q-function parameters $\{\bar{\phi}_i \leftarrow \phi_i\}_{i=1}^{M}$, polyak coefficient τ, number of critics sampled $N < M$, batch size B, number of critic updates per iteration G, replay buffer $\mathcal{D}$

for *each iteration* **do**
  Reset environment and observe initial state s_0;
  while *not terminal* **do**
    Select action $a_t \sim \pi_\theta(\cdot|s_t)$;
    Execute a_t in environment;
    Observe reward r_t, next state s_{t+1}, done signal d_t;
    Store $(s_t, a_t, r_t, s_{t+1}, d_t)$ in $\mathcal{D}$;
    if d_t *is True* **then**
      Reset environment and observe new initial state s_{t+1};
    end
    $s_t \leftarrow s_{t+1}$;
  end
  for $g = 1$ **to** G **do**
    Sample a batch of B transitions (s, a, r, s', d) from $\mathcal{D}$;
    Sample a random subset $\mathcal{I} \subset \{1, \ldots, M\}$ of size N;
    Compute target Q-values:

$$y = r + \gamma(1-d)\left(\min_{i \in \mathcal{I}} Q_{\bar{\phi}_i}(s', a') - \alpha \log \pi_{\bar{\theta}}(a'|s')\right), \quad a' \sim \pi_{\bar{\theta}}(\cdot|s')$$

    for $i = 1$ **to** M **do**
      Update critic i parameters by gradient descent:

$$\phi_i \leftarrow \phi_i - \lambda_Q \nabla_{\phi_i} \frac{1}{B} \sum \left(Q_{\phi_i}(s, a) - y\right)^2$$

      Update target critic network i:

$$\bar{\phi}_i \leftarrow \tau \phi_i + (1-\tau)\bar{\phi}_i$$

    end
  end
  Update actor parameters by gradient ascent:

$$\theta \leftarrow \theta + \lambda_\pi \nabla_\theta \frac{1}{B} \sum \left(\frac{1}{M} \sum_{i=1}^{M} Q_{\phi_i}(s, \pi_\theta(s)) - \alpha \log \pi_\theta(\pi_\theta(s)|s)\right)$$

end