Algorithm:Deep Q Learning algorithm

```
Memory ← Initialize memory container with zeros
step ← 0
Network ← Initialize weights with zeros
γ ← Discount factor
ϵ ← ϵ-greedy algorithm factor
function GET RESULT(S as state)
    for 1 ≤ i ≤ 108 do                    ▷ There are 108 cards in a UNO deck
        Mask_i ← [Card_i ∈ S.Playable]          ▷ [ ] represents Iverson bracket
    Mask_0 ← 1                          ▷ Player may always take a card from the deck
    Result ← Sigmoid(Network.Run(S)) + 1
    return (Mask ⊙ Result) - 1                ▷ ⊙ represents Hadamard product
function GET ACTION(S as state)
    if Random.Uniform(0, 1) > ϵ then
        return Dummy Algorithm.Get Action(S)
    else
        return Argmax(Get Result(S))
function OBSERVE(S as state, A as action, R as reward, S′ as state)
    Memory[step mod N] ← S, A, R, S′          ▷ N is an arbitrary number
    Batch ← Random.Batch(Batch Size)
    S, A, R, S′ ← Memory[Batch]
    Values ← Network.Run(S)[A] + R + γ · max_A′ Get Result(S′)[A′]
    Network.Run(S)[A] ← Perform gradient descent on Values
    step ← step + 1
```