# *Q*-Learning Algorithm

**APPLY-Q-POLICY**($Q$,$s$) **returns** $a$;
    $curr\text{-}Q := -\infty$;
    **for all** $a \in A$
        **if** $Q(s,a) > curr\text{-}Q$ **then**
            $curr\text{-}Q := Q(s,a)$;
            $curr\text{-}a := a$;
    **return** $curr\text{-}a$;

# *Q*-Learning Algorithm

```
BUILD-SEQUENCE(s,Q) returns Seq;
    Seq = [s];
    while LAST(Seq) ∉Terminal-State do
        a := APPLY-Q-POLICY(Q,LAST(Seq));
        APPEND(Seq,a);
        s := RESULT(APPLY(a,LAST(Seq));
        APPEND(Seq,s);
    APPEND(Seq,⊥);
    return Seq;
```

# *Q*-Learning Algorithm

**UPDATE-Q-VALUES**($Seq, Q$) **returns** $Q$;
  $\langle s,a \rangle$ = **FIRST-PAIR**($Seq$);
  **while** $a \neq \perp$ **do**
    $\langle s',a' \rangle$ = **NEXT-PAIR**($Seq$);
    **if** $a' \neq \perp$ **then**
      $Q(s,a) := (1 - \alpha)Q(s,a) +$
        $\alpha(R(s,a) + \gamma \max_{a'} Q(\langle s',a' \rangle))$;
    **else**
      $Q(s,a) := (1 - \alpha)Q(s,a) + \alpha R(s,a)$;
    $\langle s,a \rangle = \langle s',a' \rangle$
  **return** $Q$;

# SARSA Algorithm

**UPDATE-Q-VALUES**($Seq, Q$) **returns** $Q$;
    $\langle s,a \rangle$ = **FIRST-PAIR**($Seq$);
    **while** $a \neq \perp$ **do**
        $\langle s',a' \rangle$ = **NEXT-PAIR**($Seq$);
        **if** $a' \neq \perp$ **then**
            $Q(s,a) := (1 - \alpha)Q(s,a) +$
                $\alpha(R(s,a) + \gamma\, Q(\langle s',a' \rangle))$;
        **else**
            $Q(s,a) := (1 - \alpha)Q(s,a) + \alpha R(s,a)$;
        $\langle s,a \rangle = \langle s',a' \rangle$
    **return** $Q$;