

```

package rtdp;

import java.util.*;
import java.text.*;

public class RTDP extends Solver {

    public RTDP(Racetrack r) {
        super(r);
    }

    public double runTrial() {

        // Select start state at random
        Racetrack.State s = _racetrack.chooseInitialState();

        // Execute trajectory until max length reached or success
        // Note: assume no discounting (we have bounded total reward)
        double accum_reward = 0.0;
        for (int actions = 0;
            actions < Racetrack.MAX_TRAJ_LENGTH && s ctype != 'f';
            actions++) {

            // Populate neighbors
            s.populateNeighbors();

            // Bellman update and get greedy action for upper bound
            int act = s.doBellmanUpdateUpper(true /* do update */);

            // Take action and randomly pick next state
            s = s.chooseNextState(act);

            // Decrement reward for action taken
            accum_reward = accum_reward - 1.0;
        }

        return accum_reward;
    }
}

```