LearnLib has 3 interfaces:

* LearningAlgorithm -> encapsulates implementations of learning algorithms with 3 methods:
  + startLearning ->Starts the initial learning round. Returns an initial conjecture.
  + getHypothesisModel -> Returns current conjecture i.f.o. automata
  + refineHypothesis ->
    - if current conjecture is inadequate, by existence of a counterexample:
      * provides a counterexample
      * this triggers another learning round
      * process is repeated to produce an adequate model
* MembershipOracle -> Encapsulates any structure that can answer membership queries
  + processQuery() -> is provided a collection of query objects
* EquivalenceOracle ->
  + findCounterExample -> if the equivalence oracle finds a behavioral mismatch, this method returns a counterexample.

Examples

1: Constructs compactDFA and saves the inputalphabet

Has DFA SimulatorOracle and DFACounterOracle (latter is based on SimulatorOracle)

Create ExtensibleLStarDFA.withAlphabet.withOracle

construct a W-method conformance test

construct a learning experiment from

// the learning algorithm and the conformance test.

// The experiment will execute the main loop of

// active learning