**The Main Class:**

/////////////////////////////////////////////////////////////////////////////////////////////////////////////

// Major : Computer Science

// Course : Computer Theory - 601322

// Assignment No. : 2 || First Semester 2016-2017

// Author : Ibrahim Alhamad

// Author ID : 201220475

// Description : It is a program to build a dynamic Finite Automata for any language with any alphabet

// and then you can test the machine to accept or reject any given word.

/////////////////////////////////////////////////////////////////////////////////////////////////////////////

**public** **class** Main {

**public** **static** **void** main(String args[]) {

DFA dfa = **new** DFA();

**while** (**true**) {

System.***out***.print("Enter a word ( 'stop' for exit ): ");

String word = dfa.input.next();// get the word from the user.

System.***out***.println(dfa.isItAccepted(word));// check the word.

**if** (word.equals("stop")) {

dfa.input.close(); // close the inputStream.

System.*exit*(0); // exit the program.

}

}

}

}

**The DFA Class:**

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.Scanner;

**public** **class** DFA {

**private** String transitionsTable[][]; // transition table as a 2D array

**private** List<Character> Alphabet; // the Alphabet as a List Data Structure

**private** String States[]; // Array of states

**private** String FinalStates[]; // array of the final states

**private** String StartState; // the start state

**private** String currentState; // the current state

**private** List<String> statesAsList; // states array as a List

**public** Scanner input; // Scanner object

**public** DFA() {

**this**.input = **new** Scanner(System.***in***); // inputStream

**this**.Alphabet = Arrays.*asList*(fillAlphabet()); // convert the Alphabet array into a List.

**this**.States = fillStates(); // set the states array.

**this**.statesAsList = Arrays.*asList*(**this**.States); // convert the states into a List.

**this**.StartState = setStartState(); // set the start state.

**this**.FinalStates = fillFinalStates(); // set the final states array.

**this**.transitionsTable = fillTransitionsTable(); // fill the transitions table by the user.

}

**public** String isItAccepted(String token) {

**this**.currentState = **this**.StartState; // set the current state to the start state

**for**(**int** index = 0;index < token.length();index++) {

**if** (Alphabet.contains(token.charAt(index))) { // check if the current character is in the Alphabet

**int** csAsIndex = Integer.*parseInt*(**this**.currentState); // copy the currentState as an Index

**int** indexOfTheChar = **this**.Alphabet.indexOf(token.charAt(index)); // get the current char Index

**this**.currentState = **this**.transitionsTable[csAsIndex][indexOfTheChar]; // were the magic happen

} **else** {

**return** "Rejected"; // Rejected if the word contains non-alphabet characters

}

}

**for** (**int** i = 0; i < FinalStates.length; i++) {

**if** (currentState.equals(FinalStates[i])) {

**return** "Accepted"; // Accepted if the current state is one of the final states

}

}

**return** "Rejected";

}

**private** Character[] fillAlphabet() {

System.***out***.print("Enter the number of Alphabet elements: ");

**int** numOfAlphabet = 0;

**try** { numOfAlphabet = **this**.input.nextInt(); }

**catch** (java.util.InputMismatchException e) { // catch the error if the input is not a number

System.***out***.println("Invalid input!");

}

**while** (numOfAlphabet < 1) {

**try** {

System.***out***.println("The number of Alphabet cannot be < 1: ");

System.***out***.print("Enter the number of Alphabet elements: ");

numOfAlphabet = **this**.input.nextInt();

} **catch** (java.util.InputMismatchException e) { // catch the error if the input is not a number

System.***out***.print("Invalid input!\nEnter the number of Alphabet elements: ");

**this**.input.next();

}

}

Character Alphabet[] = **new** Character[numOfAlphabet]; // set the size of the Alphabet

**for** (**int** i = 0; i < Alphabet.length; i++) {

System.***out***.print("Enter the element " + (i+1) + " of the Alphabet: ");

Alphabet[i] = **this**.input.next().charAt(0);

}

System.***out***.print("This is your Alphabet: ");

System.***out***.print("{ "); // this

**for** (**int** i = 0; i < Alphabet.length; i++) { // for

**if** (i == (Alphabet.length - 1)) { // print

System.***out***.print(Alphabet[i]); // the

} **else** { // alphabet

System.***out***.print(Alphabet[i] + ", "); // to

} // the

} // user

System.***out***.println(" }");

**return** Alphabet;

}

**private** String[] fillStates() {

System.***out***.print("Enter the number of states: ");

**int** numOfStates = 0;

**try** {

numOfStates = **this**.input.nextInt();

} **catch** (java.util.InputMismatchException e) { // catch the error if the input is not a number

System.***out***.println("Invalid input!");

}

**while** (numOfStates < 1) {

**try** {

System.***out***.println("The number of States cannot be < 1: ");

numOfStates = **this**.input.nextInt();

} **catch** (java.util.InputMismatchException e) { // catch the error if the input is not a number

System.***out***.print("Invalid input!\nEnter the number of states: ");

**this**.input.next();

}

}

String States[] = **new** String[numOfStates]; // split the states string into states array.

**for** (**int** i = 0; i < States.length; i++) {

States[i] = "" + i;

}

System.***out***.print("You have these states :");

System.***out***.print("[ "); // this

**for** (**int** i = 0; i < States.length; i++) { // for

**if** (i == (States.length - 1)) { // print

System.***out***.print(States[i]); // the

} **else** { // states

System.***out***.print(States[i] + ", "); // to

} // the

} // user

System.***out***.println(" ]"); // ....

**return** States;

}

**private** String setStartState() {

System.***out***.print("Select the start state: ");

String startState = **this**.input.next(); // set the start state to the user this.input.

**while** (!**this**.statesAsList.contains(startState)) { // check if the input is one of the states?

System.***out***.println("the state you entered is not one of the states!");

System.***out***.print("Select the start state: ");

startState = **this**.input.next();

}

**return** startState;

}

**private** String[] fillFinalStates() {

System.***out***.print("Enter the number of final states: ");

**int** numOfFinalStates = -1;

**try** {

numOfFinalStates = **this**.input.nextInt();

} **catch** (java.util.InputMismatchException e) {// catch the error if the input is not a number

System.***out***.println("Invalid input!");

}

**while** (numOfFinalStates < 0) {

**try** {

System.***out***.println("The number of final States cannot be < 0: ");

numOfFinalStates = **this**.input.nextInt();

} **catch** (java.util.InputMismatchException e) {// catch the error if the input is not a number

System.***out***.print("Invalid input!\nEnter the number of final states: ");

**this**.input.next();

}

}

String FinalStates[] = **new** String[numOfFinalStates]; // split the states string into states array.

**for** (**int** i = 0; i < FinalStates.length; i++) {

System.***out***.print("select the final state (" + (i+1) + ") from states: ");

FinalStates[i] = **this**.input.next();

**while** (!**this**.statesAsList.contains(FinalStates[i])) { // check the input is one of the states?

System.***out***.println("the state you entered is not one of the states!");

System.***out***.print("Select the start state: ");

FinalStates[i] = **this**.input.next();

}

}

**return** FinalStates;

}

**private** String[][] fillTransitionsTable() {

**int** rows = States.length; // set the rows number to the state numbers

**int** cols = Alphabet.size(); // set the cols number to the alphabet numbers

String[][] data = **new** String[rows][cols]; // set the transitions table size (rows \* cols)

**for**(**int** row = 0; row < data.length; row++) { // loop on each row

**for**(**int** col = 0;col < data[row].length; col++) { // loop through each row

System.***out***.print("Transition('" + States[row] + "', " + Alphabet.get(col) + ") = ");

data[row][col] = **this**.input.next(); // Transition(x, y) = z

**while** (!**this**.statesAsList.contains(data[row][col])) { // check if the input is one of the states?

System.***out***.println("the state you entered is not one of the states!");

System.***out***.print("Transition('" + States[row] + "', " + Alphabet.get(col) + ") = ");

data[row][col] = **this**.input.next();

}

}

System.***out***.println("---------------------");

}

**return** data; // return the transitions table

}

}