/\*Isabelle Kreienbrink

\* Assignment 2 Part 2\*/

List<string> foundStates = new List<string>();

List<string> statesToCall = new List<string>();

List<string> statesCalled = new List<string>();

List<string> dfaInput = File.ReadAllLines("./DFAinput.txt").ToList();

List<List<string>> brokenUpInput = new List<List<string>>();

for (int i = 0; i < dfaInput.Count; i++)

{

List<string> individualStates = dfaInput[i].Split().ToList();

brokenUpInput.Add(individualStates);

}

List<List<string>> myDFA = NFAConversion("N0");

foreach (var list in myDFA)

{

foreach(var element in list)

Console.Write(element + " ");

Console.WriteLine("");

}

/\*

Console.WriteLine("Enter a state (just the number) that you would like to find the e-closure for");

string stateToFind = Console.ReadLine();

Console.WriteLine("The eclosure for " + stateToFind + " is:");

//Start Algorithm

Eclosure2(stateToFind);

while (statesToCall.Count > 0)

for (int i = 0; i < statesToCall.Count; i++)

{

string stateToRemove = statesToCall[i];

Eclosure2(statesToCall[i]);

statesCalled.Add(stateToRemove);

statesToCall.Remove(stateToRemove);

}

foreach (var state in foundStates)

{

Console.WriteLine(state);

}

foundStates.Clear();

statesToCall.Clear();

statesCalled.Clear();

\*/

void Eclosure2(string node)

{

int i = 0;

while (i < brokenUpInput.Count)

{

if (brokenUpInput[i][0] == node)

{

if (!foundStates.Contains(brokenUpInput[i][0]))

foundStates.Add(brokenUpInput[i][0]);

if (brokenUpInput[i][1] == "eps")

{

if (!foundStates.Contains(brokenUpInput[i][2]))

foundStates.Add(brokenUpInput[i][2]);

if (!statesToCall.Contains(brokenUpInput[i][2]))

{

if (!statesCalled.Contains(brokenUpInput[i][2]))

statesToCall.Add(brokenUpInput[i][2]);

}

}

}

i = i + 1;

}

}

List<List<string>> NFAConversion(string N0)

{

List<List<string>> toPrintNFAstates = new List<List<string>>();

Eclosure2(N0);

while (statesToCall.Count > 0)

for (int i = 0; i < statesToCall.Count; i++)

{

string stateToRemove = statesToCall[i];

Eclosure2(statesToCall[i]);

statesCalled.Add(stateToRemove);

statesToCall.Remove(stateToRemove);

}

List<string> D0 = new List<string>(foundStates);

foundStates.Clear();

statesToCall.Clear();

statesCalled.Clear();

//create algo later

List<string> allChars = new List<string> { "a", "c", "t", "o", "w" };

List<List<string>> workList = new List<List<string>>();

List<List<string>> workHistory = new List<List<string>>();

workList.Add(D0);

toPrintNFAstates.Add(D0);

List<string> newDFAstate = new List<string>();

int currStateIndex = 0;

List<string> currentStateD = workList[currStateIndex];

while (workList.Count > 0)

{

foreach (var letter in allChars)

{

foreach (string Nstate in currentStateD)

{

for (int i = 0; i < brokenUpInput.Count; i++)

{

if (brokenUpInput[i][0] == Nstate)

{

if (brokenUpInput[i][1] == letter)

{

Eclosure2(brokenUpInput[i][2]);

while (statesToCall.Count > 0)

for (int k = 0; k < statesToCall.Count; k++)

{

string stateToRemove = statesToCall[k];

Eclosure2(statesToCall[k]);

statesCalled.Add(stateToRemove);

statesToCall.Remove(stateToRemove);

}

List<string> tempList = new List<string>(foundStates);

//need to add to this, not make a new one

foreach(var item in tempList)

{

newDFAstate.Add(item);

}

foundStates.Clear();

statesToCall.Clear();

statesCalled.Clear();

/\*if (!workList.Contains(newDFAstate))

{

workList.Add(newDFAstate);

toPrintNFAstates.Add(newDFAstate);

}\*/

}

}

}

}

if (!workList.Contains(newDFAstate) && newDFAstate.Count() > 0)

{

bool isDup = false;

List<string> tempList2 = new List<string>(newDFAstate);

foreach (List<string> stringList in workHistory)

{

if (stringList.SequenceEqual(newDFAstate))

{

isDup = true;

}

}

if (!isDup)

{

workList.Add(tempList2);

toPrintNFAstates.Add(tempList2);

workHistory.Add(tempList2);

}

}

newDFAstate.Clear();

}

workList.Remove(currentStateD);

if(workList.Count > 0)

currentStateD = workList[0];

}

return toPrintNFAstates;

}