Juan Lopez

//Juan Lopez

//ECE511

#include <string>

#include <cstring>

#include <vector>

#include <iostream>

#include <fstream>

using namespace std;

struct table{

string state,newstate;

char input,output,direction;

table(string st, string inp, string nws, string outp, string dir){

state=st;

if(inp[0]=='\\')

input = '\0';

else

input = inp[0];

newstate=nws;

if(outp[0]=='\\')

output = '\0';

else

output=outp[0];

direction=dir[0];

}

void print()

{

cout << state <<", ";

if(input == '\0')

cout << "\\0" << ", ";

else

cout << input << ", ";

cout << newstate <<", ";

if(output == '\0')

cout << "\\0" << ", ";

else

cout << output <<", ";

cout << direction << endl;

}

};

void makeTape(string input, vector<char> &right){

for(int i=input.length();i>=0;i--){

right.push\_back(input[i]);

}

}

void removeSpaces(string &inp){

int count =0;

for(int j=0; inp[j] ;j++)

if(inp[j] != ' ')

inp[count++] = inp[j];

inp[count]='\0';

char \* sec = new char[count+1];

for(int i=0;i < count+1; i++)

sec[i] = inp[i];

inp = sec;

}

void makeTMTable(string inp, vector <table> &tmtable)

{

string filen=inp;

filen+=".tm";

const char \* fileName=filen.c\_str();

ifstream inputF;

inputF.open(fileName);

if(inputF.fail()){

cout << "Error, No TM file" <<endl;

exit(1);

}

string s1 = "";

int t = 0;

while(true){

string state;

getline(inputF,state,',');

removeSpaces(state);

if(state == s1)

t++;

else{

s1 = state;

t=0;}

string inp;

getline(inputF,inp,',');

removeSpaces(inp);

string nstate;

getline(inputF,nstate,',');

removeSpaces(nstate);

string op;

getline(inputF,op,',');

removeSpaces(op);

string dir;

getline(inputF,dir);

removeSpaces(dir);

if(inputF.eof()){

break;

}

tmtable.push\_back(table(state,inp,nstate,op,dir));

}

inputF.close();

}

void printRight(vector<char> right){

if(right.empty())

return;

for(int i= right.size()-1; i >0;i--)

cout << right[i];

cout << endl;

}

void printLeft(vector<char> left){

if(left.empty())

return;

for(int i=0; i < left.size(); i++)

cout << left[i];

}

void solveTaper(vector<char>&right, vector<char>&left, vector<table>&tmtable){

char currv = right.back(); //Just looking at the start state # and putting 1 into nextstate pushing the # to the left

//left because it is direction R. removes it from right and new curr = pushback();

right.pop\_back();

string currs =tmtable[0].state;

while(currs !="stop"){

cout << "State: " << currs << " Transition with input --> " << currv <<endl;

printLeft(left);

cout << "|" << currv << "|" ;

printRight(right);

cout << endl;

for(int i=0;i<tmtable.size();i++){

if(currs == tmtable[i].state)

{

if(currv == tmtable[i].input)

{

if(tmtable[i].direction == 'R')

{ left.push\_back(tmtable[i].output);

if(!right.empty()){

currv = right.back();

right.pop\_back();

}

else

currv = '\0';

}

else if(tmtable[i].direction == 'L'){

right.push\_back(tmtable[i].output);

if(!left.empty()){

currv=left.back();

left.pop\_back();

}

else

currv = '\0';

}

else if(tmtable[i].direction == '-')

currv = tmtable[i].output;

else{

cout << " Invalid Entry " << endl;

exit(1);

}

currs=tmtable[i].newstate;

break;

}

else if(tmtable[i].input=='?'){

if(tmtable[i].direction =='R'){

if(tmtable[i].output != '?'){

left.push\_back(tmtable[i].output);

if(!right.empty()){

currv = right.back();

right.pop\_back();

}

else

currv = '\0';

}

else{

left.push\_back(currv);

if(!right.empty()){

currv=right.back();

right.pop\_back();

}

else

currv = '\0';

}

}

else if (tmtable[i].direction == 'L'){

if(tmtable[i].output != '?'){

right.push\_back(tmtable[i].output);

if(!left.empty()){

currv =left.back();

left.pop\_back();

}

else

currv = '\0';

}

else {

right.push\_back(currv);

if(!left.empty()){

currv = left.back();

left.pop\_back();

}

else

currv = '\0';

}

}

else if (tmtable[i].direction == '-'){

if(tmtable[i].output != '?')

currv = tmtable[i].output;

}

else

{ cout << " Invalid Direction " << endl;

exit(1);

}

currs=tmtable[i].newstate;

break;

}

}

}

}

printLeft(left);

cout <<currv;

printRight(right);

cout <<endl;

}

int main(int argc,char \*argv[]){

vector<char>right;

vector<char>left;

vector<table>tmtable;

if(argc<3){

cout << "Error, Not enough arguements, argv[0] = a.out, argv[1] = the tm file, argv[2] = the input " <<endl;

exit(1);

}

makeTMTable(argv[1],tmtable);

makeTape(argv[2],right);

for(int i =0; i< tmtable.size();i++)

tmtable[i].print();

printRight(right);

solveTaper(right,left,tmtable);

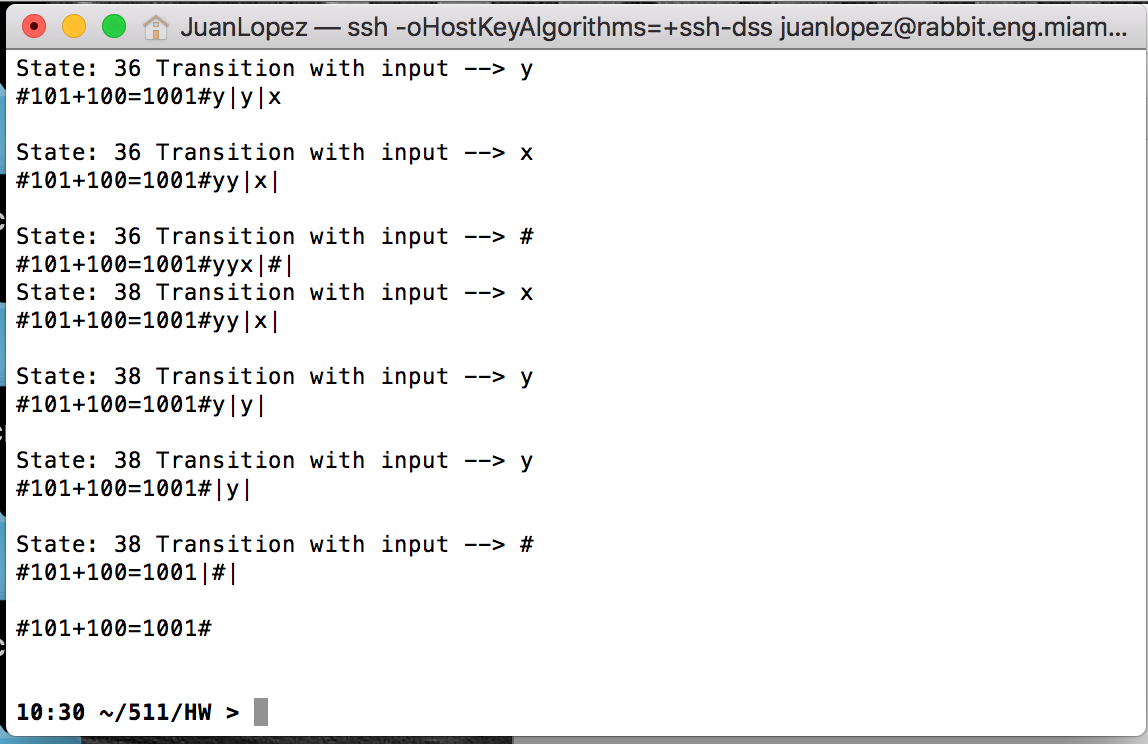
cout << endl;

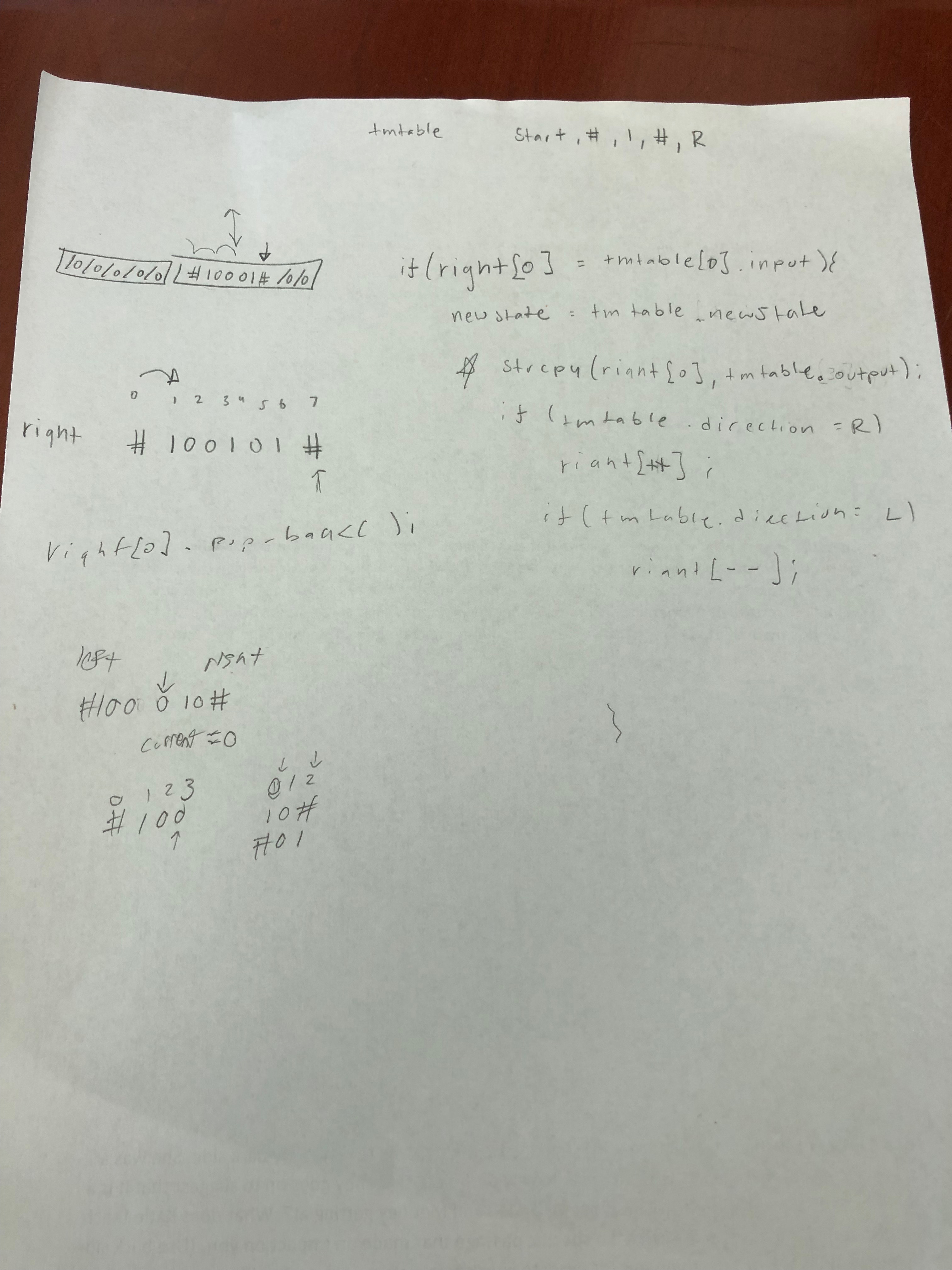
return 0;

}

As much as this was a very tedious assignment I enjoyed working through it and facing a huge battle which was my own mind.

Thank you. Professor Murrell





Also wanted to Attach how I came to this solution

