<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.2/jquery.min.js"></script>

<script>

$(window).ready(function(){

var debug = false;

var input = $("#calc\_tb");

var $posfix = $("#posfix");

var $answer = $("#answer");

var bFirstTime = true;

var $results = $("#results");

var bWasNumeric =false;

var bisNumeric = false;

var $rb =$("#radians\_rb");

// if(debug)$results.html('1 +sin(30) + cos(60) + 2.1')

var functions = [ 'sqrt' , 'cos' , 'sin' , 'log10', 'ln' ,'abs' ,'tan' ]

function isOperand(elem, bAllowParenthesis){

elem = $.trim( elem) ;

if(bAllowParenthesis && ( elem == '(' || elem == ')' ) )

return true;

return (elem == '+' || elem == '-' || elem == '\*' || elem =='/' || elem == '^' );

}

function displayError(msg, data)

{

var h =''

if(msg =='mismatched\_parenthesis')

h = ' Your paranthesis are not balanced';

else if(msg == 'too\_many\_decimals')

h = ('You entered a number with too many decimals');

else if(msg == 'number\_parse\_problem')

h = ('We had trouble understanding the following term ' + data);

else if(msg == 'rpn\_pop\_pop')

h = ('We we were unable to process your expression') //This postifix syntax is invalid');

else if (msg == 'division\_by\_zero' )

h = (' You are trying to divide by zero!' );

else if (msg =='could\_not\_parse')

h = ('We we were unable to process your expression');

$("#errmsg").css('display','block');

$("#errmsg").html( h )

}

function getFullNumber(str){

var bFoundDot = false;

var out="";

for( var j = 0 ;j< str.length ; j++)

{

var currLett = str.substring(j, j + 1 ) ;

if( '0123456789'.indexOf(currLett) != -1 )

out += currLett;

else if( '.' == currLett && bFoundDot)

{

displayError('too\_many\_decimals');

break;

}

else if( '.' == currLett && bFoundDot== false)

{

out += currLett ;

bFoundDot = true;

}

else

break;

}

return out;

}

function toArr(data){

var bAllowParenthesis = true;

var arr =[];

for(var i = 0;i< data.length; i++)

{

var currLett = data.substring(i, i+1);

if(currLett== ' ') continue;

//is a dot or is a digit

var bisNumeric = currLett == '.' || '0123456789'.indexOf(currLett) != -1 ;

if(bisNumeric )

{//get the full number which could be several characters long

var fullNum = getFullNumber(data.substring(i ) )

var isOk = isNaN( +fullNum) == false; //one last check that it actually is a number

if(isOk )

arr.push( parseFloat( fullNum) +"") ;

else

displayError('number\_parse\_problem', fullNum);

i += fullNum.length-1; // -1 because i is i++ in loop

}

else if( isOperand( currLett, bAllowParenthesis) )

arr.push(currLett);

}

//there's got be a better way to do this..but it works 😉

//support unary negative signs by finding them and putting them with numbers they preceed

for( i = arr.length-1 ;i >= 0 ; i--)

{

var token = arr[i];

//look for negatives

if( arr[i] == '-' )

{

if( i ==0 && arr.length > 0)

{

arr[0] = '-' + arr[1];

arr.splice(1,1);

}

//negative preceded by an operand and is a unary

else if ( i+ 1< data.length && i > 0 && isOperand(arr[i-1] , bAllowParenthesis ) && '0123456789.'.indexOf(arr[i+1] ))

{

arr[i] = '-'+ arr[i+1] ;

arr.splice(i+1,1);

}

}

}

return arr

}

function resetErr(){

$("#errmsg").css('display','none')

}

input.focus(function(){

if(bFirstTime)

input.val('')

input.css('color','black');

bFirstTime = false;

})

function p( str){if (debug) console.log(str ) ; }

function infixToPostfix(array){

var i, operandStack = [];

var output=[] ;

var bAllowParenthesis = true;

for( i = 0 ; i < array.length; i++)

{

var currentToken = $.trim( array[i] );

if( isOperand( currentToken, bAllowParenthesis ))

{

p("I is operand " + currentToken + ', output : ' +output + ", operandStack: " +operandStack);

if( operandStack.length == 0 )

operandStack.push( currentToken );

else if ( operandStack.length > 0 && currentToken == ')' )

{

while (operandStack.length > 0 && operandStack[operandStack.length-1] != '(' )

{

output. push( operandStack.pop());

}

p('\t B now, pop off ' +operandStack[operandStack.length-1] + ' SHOULD BE CLOSING PARENTHESIS!' );

if(operandStack[operandStack.length-1] != '(')

{

displayError('mismatched\_parenthesis');

return;

}

operandStack.pop() ; // remove parenthesis

}//

else if( operandStack.length > 0 )

{

p("II is operand " + currentToken + ', output : ' +output);

if( (operandStack[operandStack.length-1] == "(" && currentToken == "(" || ( currentToken != "(" ) && operatorToPrecedence(operandStack[operandStack.length-1]) >= operatorToPrecedence( currentToken) ) )

{

p(" C , operandStack : "+operandStack );

while (operandStack.length > 0 && operandStack[operandStack.length-1] != "("

&& operatorToPrecedence(operandStack[operandStack.length-1]) >= operatorToPrecedence( currentToken) )

{

output.push( operandStack.pop() );

}

p('\t D now, pop off ' +operandStack[operandStack.length-1] );

operandStack.push(currentToken) ;

}

else if ( operatorToPrecedence( operandStack[operandStack.length-1] ) < operatorToPrecedence( currentToken) )

{

p('\t III operandStack[operandStack.length-1] ,' +operandStack[operandStack.length-1] + "< " + currentToken );

operandStack.push( currentToken );

}

}

}

// ELSE IT IS ANUMBER

else if ( isNaN( +currentToken) == false) // ie it is a number

{

p("IV isNumber() currentToken = " +currentToken);

output.push(currentToken)

}

}

while( operandStack.length > 0 )

output.push( operandStack.pop() ) ;

return output;

}

function operatorToPrecedence( op){

if( op == "+" || op == "-" )

return 1;

else if( op== "\*" || op == '/')

return 2;

else if (op == '^')

return 3;

else if (op == '(' || op == ')')

return 4;

else

throw ("Unknown operator =" +op + ',at operatorToPrecedence()')

}

///this does not actually us a stack because the user has entered teh entire equation

//

function evaluateRPN( rpnArray )

{

var operandsStack = [];

var r = 0;

var i =0;

var iterationCount = 0;

while (rpnArray.length > 1)

{

var currentToken = $.trim(rpnArray[i]) ;

if( isOperand( currentToken ))

{ //console.log(' isOperand() ' + op);

var op = rpnArray.splice(i,1);

var insertAt = i-2;

i--;

if(rpnArray.length < 2)

{

displayError('rpn\_pop\_pop')

return;

}

var n1Was = rpnArray.splice(i,1);

i--;

var n2Was = rpnArray.splice(i,1);

var n1 =parseFloat(n1Was );

var n2 =parseFloat( n2Was);

if( isNaN( +n1))

{

console.log('n1 , ' + n1Was + ', is not a number. Parsing exiting now');

displayError('could\_not\_parse');//invalid\_token\_rpn', 'Invalid syntax. ' + n1Was + ' should be a number');

return ;

}

if( isNaN( +n2))

{

displayError('could\_not\_parse');

console.log('n2 , ' + n2Was + ', is not a number. Parsing exiting now ');

return ;

}

var pushMe = calculate( n2, n1, op);

rpnArray.splice(insertAt, 0, pushMe );

}

else

i++;

if (iterationCount++ > 500 )

{

displayError('could\_not\_parse');

console.log('get me outta here, there is something wrong');

return;

}

}

if(rpnArray.length != 1)

{

displayError('could\_not\_parse');

console.log('unable to parse postfix expression : ' + rpnArray.toString().substring(1, rpnArray.toString().length-1) );

}

return rpnArray.pop();

}

function calculate (a,b,op){

if( op == '+')

return a + b;

else if( op == '-')

return a - b;

else if( op == '\*')

return a \* b;

else if( op == '^')

return Math.pow(a,b);

else if( op == '/')

{

if(b == 0)

{

displayError('division\_by\_zero');

//throw ("division by zero");

return;

}

return a / b;

}

}

function replaceAllPis(){

var data = new String($results.text()) ;

if(data.indexOf('pi') == -1)

return;

while( data.indexOf('pi') != -1)

{

var firstPart = data.substring(0,data.indexOf('pi') );

var middle = Math.PI;

var lastPart = data.substring( data.indexOf('pi') + 2 )//

data = firstPart + middle + lastPart;

}

$results.html(data)

}

function evaluateFuncts(){

var finalString='';

var bRadians = $rb.is(":checked");

var radDeg = bRadians? 1 : ( Math.PI/180 );

replaceAllPis();

for(var i = 0 ; i < functions.length ; i++)

{

var data = new String($results.text()) ;

var fxn= functions[i];

var c =0;

var firstRun = true ;

while(data.indexOf(fxn ) != -1 && c++ < 5)

{

if(firstRun )

{

p('fxn: ' + fxn + ', data : ' +data);

firstRun =false;

}

var inject ='inject'

var iStart = data.indexOf(fxn );

var temp = data.substring(iStart + fxn.length );

var iEnd = temp.indexOf( ')' ) +1;

var lastPart = temp.substring(iEnd);;

temp = temp.substring( 0 ,iEnd); // this is ( 60) for something like cos(60)

var number = temp.substring(temp.indexOf('(')+1 , temp.indexOf(')' ));

if( isNaN( +number))

console.log('problem parsing number = ' +number);

number = parseFloat (number);

if(fxn == 'sin')

{

console.log('radDeg \* number ' +(radDeg \* number));

inject = Math.sin(radDeg \* number);

console.log('Math.sin(radDeg \* number) Math.sin(' + (radDeg \* number) +'😞 ' +inject );

}

else if(fxn == 'cos')

inject = Math.cos(radDeg \* number);

else if(fxn == 'tan')

inject = Math.tan(radDeg \* number);

else if(fxn == 'sqrt')

inject = Math.sqrt(number);

else if(fxn == 'log10')

inject = Math.LOG10E(number);

else if(fxn == 'ln')

inject = Math.LN10(number);

else if(fxn == 'abs')

inject = Math.abs(number);

var firstPart = data.substring(0,iStart);

data = firstPart + ' ' + inject + ' ' + lastPart;

$results.text( data )

}

}

}

$(".calcbttn").click(function(){

resetErr();

bisNumeric =false;

var val = $(this).val()

var inject=''

if( $.inArray (val, functions) != -1 )

{

inject = ( bWasNumeric ? ' \* ' : '' ) + val +'('

}

else if( val == 'back')

{

{

var data = new String($results.text()) ;

if(data.length == 0 )

return;

else if(data.length == 1 && data == ' ')

return;

var lastLttr = data.substring(data.length-1 );

if(lastLttr == ' ')

data = data.substring(0, data.length-2) ;

else

data = data.substring(0, data.length-1) ;

$results.html (data);

return;

}

}

else if(val == 'clear')

{

$results.html('')

return;

}

else if (val == 'parenleft')

inject = '(' ;

else if (val == 'parenright')

inject = ')' ;

else if(val== '=')

{

evaluateFuncts();

var arr = toArr($results.text());

arr = infixToPostfix(arr) ;

var theResult= evaluateRPN(arr);

$results .html (theResult);

return;

}

else if( val == 'pi')

{

inject = 'pi'

bisNumeric = true;

}

else

{

if( isNaN(+val ) == false || val == '.')

bisNumeric = true;

inject = val;

}

// console.log( $(this).val() + ' was clicked ');

var space = (bWasNumeric && bisNumeric ) ? '' : ' ';

$results .html ( $results.html() + space + inject );

bWasNumeric = bisNumeric;

})

$("#clear").click(function(){

input.val('');

})

})

</script>

<!-- add a little style to the calculator-->

<style>

</head>

<body>

\*{padding:0;

margin:0;

}

body {text-align:center;

background-color:brown;

}

.calc{

border:5px solid black;

margin-left:450px;

margin-right:450px;

margin-top:20px;

padding:10px;

background-color:gray;

}

.calc h1{

text-align:center;

color:white;

}

input{

font-size:15px;

font-weight:bold;

padding:5px;

margin:5px;

}

#scientificCalc11 {

font-size: 15px;

margin:0;

color:#666;

text-shadow: 1px 1px 5px #ffffff;

font-weight:bold;

font-family: "MuseoSans500Regular", sans-serif;

border:2px;

}

#scientificCalc11 cell{

width:20%;

font-weight:bold

}

#scientificCalc11 #results{

background-color:#D5F192;

height:40px;

width:373px;

margin:5px;

padding:13px;

border:2px;

}

.calcbttn {

border:5px;

cursor:pointer;

font-family: "MuseoSans500Regular", sans-serif;

text-shadow: 1px 1px 2px #ffffff;

color:;

font-size: 20px;

border:2px;

background-repeat: no-repeat;

background: #eee;

padding:5px;

height:50px;

width:70px;

</style>

<center>

<!-- now the html part -->

<div id="scientificCalc11">

<div class="calc">

<h1>SCIENTIFIC CALCULATOR</h1>

<div id="results"></div>

<table style="width:350px;">

<tr>

<td class="cell">

<input type="button" class='calcbttn' value="7" id="7" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="8" id="8" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="9" id="9" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="/" id="divide" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="(" id="parenleft" /></td>

</tr>

<tr>

<td class="cell">

<input type="button" class='calcbttn' value="4" id="4" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="5" id="5" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="6" id="6" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="\*" id="multiply" /></td>

<td class="cell">

<input type="button" class='calcbttn' value=")" id="parenright" /></td>

</tr>

<tr>

<td class="cell">

<input type="button" class='calcbttn' value="1" id="1" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="2" id="2" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="3" id="3" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="+" id="plus" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="sqrt" id="sqrt" />

</td>

</tr>

<tr>

<td class="cell">

<input type="button" class='calcbttn' value="0" id="0" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="." id="decimal"/></td>

<td class="cell">

<input type="button" class='calcbttn' value="=" id="evaluate" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="-" id="minus" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="^" id="^" /></td>

</tr>

<tr>

<td class="cell">

<input type="button" class='calcbttn' value="abs" id="abs" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="sin" id="sin" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="cos" id="cos" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="tan" id="tan" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="back" id="del" /></td>

</tr>

<tr>

<td class="cell">

<input type="button" class='calcbttn' value="log10" id="log10"/></td>

<td class="cell">

<input type="button" class='calcbttn' value="log2" id="log2" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="ln" id="ln" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="pi" id="pi" /></td>

<td class="cell">

<input type="button" class='calcbttn' value="clear" id="clear"/></td>

</tr>

</table>

<input type="radio" name="raddeg" value="radians" id="radians\_rb"/> Radians <input type="radio" name="raddeg" value="degrees" checked id="degrees\_rb" /> degrees

</div>

<!-- REMOVING THE TEXT BELOW VIOLATES THE TOS OF USING THIS SCRIPT -->

<span style="font-size:12px; font-style:italic;"></span>

</center>

</body>

</html>