===========================================================================================================

CODIGO NA TIMELINE ROOT

===========================================================================================================

var id\_ia:Number;

var ia\_ligado:Boolean = false;

var ultimoMovido:Number = -1;

var configTab:Array = [];

// resolve aleatoriamente

function aleatorio() {

var blocosMoviveis:Array = [];

for(i=1;i<10;i++) {

if(tabuleiro.movivel(i) && i != ultimoMovido)

blocosMoviveis.push(i);

}

// seleciona aleatoriamente um dos blocos moviveis

id = blocosMoviveis[int((Math.random()\*10)%(blocosMoviveis.length))];

// move o bloco selecionado

ultimoMovido = id;

tabuleiro.mover(id);

};

// calcula soma das distancias das posicoes

function distanciaGeralModular(tabul:Array):Number {

var soma:Number = 0;

for(k=0;k<9;k++) {

dy = Math.abs(int((tabul[k]-1)/3))-Math.abs(int(k/3));

dx = Math.abs((tabul[k]-1)%3)-Math.abs(k%3);

soma += Math.sqrt(dx\*dx+dy\*dy);

}

return soma;

}

// calcula a soma das distancias das posicoes com seus valores

function distanciaGeral(tabul:Array):Number {

var soma:Number = 0;

for(k=1;k<=9;k++) {

d = Math.abs(k-tabul[k-1]);

soma += d\*d;

}

return Math.sqrt(soma);

}

// dada uma configuracao de tabuleiro e a peca que foi supostamente movida, calcula as distancias

function arvoreJogadasLv2(tab:Array,blocoMovido:Number):Number {

p1=tab[0]; p2=tab[1]; p3=tab[2]; p4=tab[3]; p5=tab[4]; p6=tab[5]; p7=tab[6]; p8=tab[7]; p9=tab[8];

var distancia:Number = 200;

// para todas as pecas

for(i=1;i<9;i++) {

// se ela for movivel...

if(\_root.tabuleiro.movivel(i) && i != blocoMovido) {

// obtem a posicao da casa vazia

for(j=0;j<9;j++) {

if(tab[j] == '9') {

posicaoVazia = j; break;

}

}

// obtem a posicao do movivel

for(j=0;j<9;j++) {

if(tab[j] == i) {

posicaoMovivel = j; break;

}

}

// troca

tab[posicaoVazia] = i;

tab[posicaoMovivel] = '9';

if(\_root.distanciaGeral(tab) < distancia) {

distancia = \_root.distanciaGeral(tab);

}

}

tab[0] = p1; tab[1] = p2; tab[2] = p3; tab[3] = p4; tab[4] = p5; tab[5] = p6; tab[6] = p7; tab[7] = p8; tab[8] = p9;

}

return distancia;

};

// dada uma configuracao de tabuleiro, calcula a soma das distancias

function arvoreJogadas(tab:Array, tipo:Number):Number {

p1=tab[0]; p2=tab[1]; p3=tab[2]; p4=tab[3]; p5=tab[4]; p6=tab[5]; p7=tab[6]; p8=tab[7]; p9=tab[8];

var distancia:Number = 200;

var melhorBloco:Number = -1;

// para todas as pecas

for(m=1;m<9;m++) {

// se ela for movivel...

if(\_root.tabuleiro.movivel(m)) {

// obtem a posicao da casa vazia

for(j=0;j<9;j++) {

if(tab[j] == '9') {

posicaoVazia = j; break;

}

}

// obtem a posicao do movivel

for(j=0;j<9;j++) {

if(tab[j] == m) {

posicaoMovivel = j; break;

}

}

// troca

tab[posicaoVazia] = m;

tab[posicaoMovivel] = '9';

dg = \_root.distanciaGeral(tab);

dgm = \_root.distanciaGeralModular(tab);

if(tipo == 1) {

if(dg < distancia) {

distancia = dg

melhorBloco = m;

}

}

else if(tipo == 2) {

// informa a peca que seria movida e a menor distancia gerada pela sua sequencia

if(\_root.arvoreJogadasLv2(tab,m)+dg < distancia) {

distancia = \_root.arvoreJogadasLv2(tab,m)+dg;

melhorBloco = m;

}

}

else if(tipo == 3) {

if(dgm < distancia) {

distancia = dgm;

melhorBloco = m;

}

}

}

tab[0] = p1; tab[1] = p2; tab[2] = p3; tab[3] = p4; tab[4] = p5; tab[5] = p6; tab[6] = p7; tab[7] = p8; tab[8] = p9;

}

return melhorBloco;

};

// verifica se uma configuracao ja foi jogada

function configRepetida(tabAtual:Array, pecaMover:Number):Boolean {

var posicaoVazia:Number;

var posicaoMover:Number;

// constroi um tabuleiro com as configuracoes se tivesse movido a peca

var tabAtualLocal:Array = [];

for(j=0;j<9;j++) {

if(tabAtual[j] == '9') {

posicaoVazia = j; break;

}

}

for(j=0;j<9;j++) {

if(tabAtual[j] == pecaMover) {

posicaoMover = j; break;

}

}

for(j=0;j<9;j++) tabAtualLocal[j] = tabAtual[j];

tabAtualLocal[posicaoVazia] = tabAtualLocal[posicaoMover];

tabAtualLocal[posicaoMover] = '9';

// verifica se ja existe ocorrencia

for(i=0;i<configTab.length;i+=9) {

for(j=0;j<9;j++) {

if(configTab[i+j] != tabAtualLocal[j])

break;

}

if(j == 9) {

vetorRecorte = [];

for(k=0;k<i;k++) vetorRecorte.push(configTab[k]);

for(k=i+9;k<configTab.length;k++) vetorRecorte.push(configTab[k]);

configTab = vetorRecorte;

return true;

}

}

return false;

};

// resolve heuristicamente

function heuristica(tipo:Number) {

// encontra a melhor opcao de movimento

var casasAtuais:Array = [];

var array:Array = [];

for(i=1;i<=9;i++) array.push(tabuleiro.posicao(i));

for(i=1;i<=9;i++) casasAtuais[array[i-1]-1] = i;

id = arvoreJogadas(casasAtuais,tipo);

// adiciona a configuracao atual do tabuleiro na pilha

if(antiloop.selected) {

casasAtuais = []; array = [];

for(i=1;i<=9;i++) array.push(tabuleiro.posicao(i));

for(i=1;i<=9;i++) casasAtuais[array[i-1]-1] = i;

for(i=0;i<9;i++) configTab.push(casasAtuais[i]);

}

// verifica se o ultimo movimento foi o escolhido

if(id == ultimoMovido && !antiloop.selected) {

escolheOutro = [];

for(i=1;i<9;i++)

if(\_root.tabuleiro.movivel(i) && i != id)

escolheOutro.push(i);

id = escolheOutro[int((Math.random()\*10)%(escolheOutro.length))];

}

// verifica loopings

if(antiloop.selected) {

if(casasAtuais.length > 0 && configRepetida(casasAtuais,id)) {

escolheOutro = [];

for(i=1;i<9;i++)

if(\_root.tabuleiro.movivel(i) && i != id)

escolheOutro.push(i);

id = escolheOutro[int((Math.random()\*10)%(escolheOutro.length))];

}

}

ultimoMovido = id;

\_root.tabuleiro.mover(id);

};

===========================================================================================================

===========================================================================================================

CODIGO DOS BOTOES

===========================================================================================================

on(click) {

var array:Array = new Array();

var embaralhamento:Array = [0,1,2,3,4,5,6,7,8];

var total = embaralhamento.length;

for(i=0;i<total;i++) {

var ran = Math.floor(Math.random()\*((embaralhamento.length-1)+1));

array.push(embaralhamento[ran]);

embaralhamento.splice(ran,1);

}

embaralhamento = array;

for(i=1;i<10;i++) {

eval("\_root.tabuleiro.b"+i).\_x = 100\*(embaralhamento[i-1]%3);

eval("\_root.tabuleiro.b"+i).\_y = 100\*(int(embaralhamento[i-1]/3));

}

\_root.contador.text = 0;

\_root.tabuleiro.contador = 0;

\_root.tabuleiro.b9.\_alpha = 0;

\_root.ultimoMovido = -1;

\_root.configTab = [];

\_root.ia\_ligado = false;

clearInterval(\_root.id\_ia);

}

on(click) {

if(\_root.ia\_ligado == false) {

\_root.ia\_ligado = true;

\_root.id\_ia = setInterval(\_root.aleatorio,int(1000/(\_root.velocidade.value\*\_root.velocidade.value\*\_root.velocidade.value)));

}

}

on(click) {

if(\_root.ia\_ligado == false) {

\_root.ia\_ligado = true;

\_root.id\_ia = setInterval(\_root.heuristica,int(1000/(\_root.velocidade.value\*\_root.velocidade.value\*\_root.velocidade.value)),1);

}

}

on(click) {

if(\_root.ia\_ligado == false) {

\_root.ia\_ligado = true;

\_root.id\_ia = setInterval(\_root.heuristica,int(1000/(\_root.velocidade.value\*\_root.velocidade.value\*\_root.velocidade.value)),2);

}

}

on(click) {

if(\_root.ia\_ligado == false) {

\_root.ia\_ligado = true;

\_root.id\_ia = setInterval(\_root.heuristica,int(1000/(\_root.velocidade.value\*\_root.velocidade.value\*\_root.velocidade.value)),3);

}

}

on(click) {

fscommand("quit",true);

}

on(release) {

if(\_root.ia\_ligado == false) {

\_parent.mover(substring(this.\_name,2,2));

}

}

===========================================================================================================

===========================================================================================================

CODIGO NA TIMELINE DO TABULEIRO

===========================================================================================================

var contador:Number = 0;

function posicao(id:String):Number {

for(i=1;i<=9;i++)

if(substring(eval("b"+i).\_name,2,2) == id)

return int(eval("b"+i).\_x%3)+int(eval("b"+i).\_y%3)\*3+1;

};

function resolvido():Boolean {

for(i=1;i<9;i++) {

if(posicao(i) != i)

return false;

}

clearInterval(\_root.id\_ia);

\_root.ia\_ligado = false;

return true;

};

function movivel(id:String):Boolean {

pos\_x = (posicao(id)-1)%3;

pos\_y = int((posicao(id)-1)/3);

pos\_xv = (posicao('9')-1)%3;

pos\_yv = int((posicao('9')-1)/3);

if(pos\_x-1 == pos\_xv || pos\_x+1 == pos\_xv)

if(pos\_y == pos\_yv)

return true;

if(pos\_y-1 == pos\_yv || pos\_y+1 == pos\_yv)

if(pos\_x == pos\_xv)

return true;

return false;

};

function mover(id:String) {

if(!resolvido()) {

if(id != '9' && movivel(id)) {

// troca as posicoes

pos\_x = eval("b"+id).\_x;

pos\_y = eval("b"+id).\_y;

eval("b"+id).\_x = eval("b"+9).\_x;

eval("b"+id).\_y = eval("b"+9).\_y;

eval("b"+9).\_x = pos\_x;

eval("b"+9).\_y = pos\_y;

// incrementa o contador

contador++;

\_root.contador.text = this.contador;

// verifica se esta resolvido

if(resolvido()) b9.\_alpha = 100;

}

}

};

===========================================================================================================