

```

//+-----+
//|                                     fimathe.mq5 |
//|                                     Copyright 2024, MetaQuotes Ltd. |
//|                                     https://www.mql5.com |
//+-----+
//+-----+
//+-----+
//| experimento.mq5 |
//| Copyright 2023, MetaQuotes Ltd. |
//| https://www.mql5.com |
//+-----+
#property copyright "Copyright 2023, MetaQuotes Ltd."
#property link "https://www.mql5.com"
#property version "1.00"
//+-----+
//| Expert initialization function |
//+-----+
#include <trade\trade.mqh>
CTrade trade;

ulong BilheteDeVenda = 0; // Inicializado com 0 para indicar que não há ordem de venda aberta
ulong BilheteDeCompra = 0; // Inicializado com 0 para indicar que não há ordem de compra aberta
ulong ticketvenda = 0; // Inicializado com 0 para indicar que não há ordem de venda aberta
ulong ticketcompra = 0; // Inicializado com 0 para indicar que não há ordem de compra

input bool GERENCIAMENTO_DE_RISCO = true;
input double porcentos = 2.0 ; // limite de perdas em porcentagem
input double poercentosg = 2.0 ; // limite de ganho em porcentagem
bool input ciclos = false;
input bool surfada = false;
datetime target_time = D'2023.12.05 20:10:00';
input int MagicNumber = 12345; // Unique identifier for the EA
input string password = "jossias"; // password
input ENUM_TIMEFRAMES PerodoOperacional = PERIOD_M15 ; // Perodo Operacional
input bool ativar_ou_desativar_venda = true; // ativar_ou_desativar_venda
input bool ativar_ou_desativar_compra = true; // ativar_ou_desativar_compra
input double compra ; // preço de compra
input double venda ; // preço de venda
input double lot = 0.01; // lot
input double santo; // pontos para fora da caixa
input bool Modificar_SI_Para_Ox0 = true; // colocar a ordem no 0
input double dedo = 0; // pontos para rompimento da vela
input bool condicao_De_rompimento_v = true; // abrir a condição para rompimento da vela na venda
input bool condicao_De_rompimento_c = true; // abrir a condição para rompimento da vela na compra
input bool Costura = false;
input int Nives = 1;
double fora ; // variavel para santo
double pc; // preço de compra pra linha
double pv; // preço de venda pra linha
double PrecoDeCompra; // preço de compra pra ordem
double PrecoDeVenda; // preço de venda pra ordem
double var1; // possível preço de compra
double var2; // possível preço de venda
double Pontos; // pontos para linhas
double divisao; // divisão
double Buytake; // take de compra
double Buystop ; // estop de compra
double Selltake ; // teke de venda
double Sellestop ; //estop de venda
double SellModif ; // modificar estop sell
double Buymodif; // modificar estop buy
double Buysubsicul; // subsicol de compra
double Sellsubsicul ; // subsicol de venda
double preco_de_abertura_de_venda ; // preço de abertura de veda
double preco_de_abertura_de_compra ; // preço de abertura de compra
double nivelzerobuy; //preco sem perda para compra
double nivelzerosell ; // preço sem perda para venda
double nivelstoplos_sell; // preço de stop loss de venda
double nivelstoplos_buy; // preco de stop loss de compra
double linhasell; // criar linha de venda para linha
double linhabuy; // criar linha de compra para linha
double linhasubsiculv; // criar linha de subsicol de venda
double linhasubsiculc; //criar linha de subsicol de compra
double pontosf; // pontos para ordem
double takbuy ; // nivel de take buy
double taksell; // nivel de take sell
double tamanho_DA_vela_para_compra = 0.0;
double tamanho_DA_vela_para_venda = 0.0;

//+-----+
//| |
//+-----+
double precosArray [];
```

```

//+-----+
//| |
//+-----+
//+-----+
//| controles |

//+-----+
bool controlbuy = false; // control de ordem de compra
bool controlsell = false; // control de ordem de venda
bool controlbuymdf = false; // control de modificação de ordem de compra
bool controlsellmdf = false; // control de modificação de ordem de venda
bool linhasc = false; // control de linha de compra
bool linhasv = false; // control de linha de venda
bool segundo_control_de_taksell = false;
bool segundo_control_de_takbuy = false ;
bool controlsurfada1 = false;
bool controlsurfada2 = false;
bool control_de_venda = false; // segundo control para venda
bool control_de_compra = false; // segundo control para compra
bool control_de_venda2 = false;
bool control_de_compra2 = false;
bool ctr1 = true; // fechar a venda ou abrir t/f
bool ctr2 = true;
bool ctr3 = true;
bool ctr4 = true;
bool ctr5 = true;
bool ctr6 = true;
bool ctr8 = true;
bool ctr7 = true;
bool ctr9 = true;
bool ctr10 = true;
bool ctr11 = true;
bool ctr12 = true;
bool ctr13 = true;
bool ctr14 = true;
bool ctr15 = true;
bool ctr16 = true;
bool ctr17 = true;
bool ctr18 = true;
bool ctr19 = true;
bool ctr20 = true;
bool ctr21 = true;
bool ctr22 = true;
bool tmp_placar = true; //tmp_placar = true exibe no comment o resultado das negociações do dia

//+-----+
//| |
//+-----+
void criarlinhascompra(string precocompra, double valor)
{
    ObjectCreate(0, precocompra, OBJ_HLINE, 0, 0, valor);
    ObjectSetInteger(0, precocompra, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, precocompra, OBJPROP_COLOR, clrYellow);
}
//+-----+
void criarlinhavenda(string precovanda, double valor)
{
    ObjectCreate(0, precovanda, OBJ_HLINE, 0, 0, valor);
    ObjectSetInteger(0, precovanda, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, precovanda, OBJPROP_COLOR, clrYellow);
}
//
void criarlinha_segundo_siclo(string siclo2, double segundo_siclo)
{
    ObjectCreate(0, siclo2, OBJ_HLINE, 0, 0, segundo_siclo);
    ObjectSetInteger(0, siclo2, OBJPROP_STYLE, STYLE_DASHDOTDOT);
    ObjectSetInteger(0, siclo2, OBJPROP_COLOR, clrWhite);
}
//+-----+
//| |
//+-----+
void criarlinha_terceiro_siclo(string siclo3, double terceiro_siclo)
{
    ObjectCreate(0, siclo3, OBJ_HLINE, 0, 0, terceiro_siclo);
    ObjectSetInteger(0, siclo3, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, siclo3, OBJPROP_COLOR, clrWhite);
}
//+-----+
//| |
//+-----+

```

```

void criarlinha_quarto_siclo(string siclo4, double quarto_siclo)
{
    ObjectCreate(0, siclo4, OBJ_HLINE, 0, 0, quarto_siclo);
    ObjectSetInteger(0, siclo4, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, siclo4, OBJPROP_COLOR, clrWhite);
}
//+-----+
//|
//+-----+
void criarlinha_2_siclo(string siclo2v, double segundo)
{
    ObjectCreate(0, siclo2v, OBJ_HLINE, 0, 0, segundo);
    ObjectSetInteger(0, siclo2v, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, siclo2v, OBJPROP_COLOR, clrWhite);
}
//+-----+
//|
//+-----+
void criarlinha_3_siclo(string siclo3v, double terceiro)
{
    ObjectCreate(0, siclo3v, OBJ_HLINE, 0, 0, terceiro);
    ObjectSetInteger(0, siclo3v, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, siclo3v, OBJPROP_COLOR, clrWhite);
}
//+-----+
//|
//+-----+
void criarlinha_4_siclo(string siclo4v, double quarto)
{
    ObjectCreate(0, siclo4v, OBJ_HLINE, 0, 0, quarto);
    ObjectSetInteger(0, siclo4v, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, siclo4v, OBJPROP_COLOR, clrWhite);
}
//+-----+
//|
//+-----+
int OnInit()
{
    if(password == "jossias")
    {
    }
    else
    {
        ExpertRemove();
    }

    var1 = compra;
    var2 = venda;
    PrecioDeCompra = var1;
    PrecioDeVenda = var2;
    pontosf = var1 - var2;
    divisao = pontosf / 2;
    Buytake = PrecioDeCompra + (pontosf * Nives);
    Buystop = PrecioDeCompra - pontosf;
    Buysubsicul = PrecioDeCompra + divisao;
    Buymodif = PrecioDeCompra;
    Selltake = PrecioDeVenda - (pontosf * Nives);
    Sellestop = PrecioDeVenda + pontosf;
    Sellsubsicul = PrecioDeVenda - divisao;
    SellModif = PrecioDeVenda;

    // linhas de siclos de alta

    //+-----+
    //|
    //+-----+

    // linhas de siclos de baixa

    //+-----+
    //|
    //+-----+

```

```

//+-----+
//|
//+-----+

double santinho = SymbolInfoDouble(_Symbol, SYMBOL_POINT);
fora = santinho * santo ;
// Definir o símbolo e o período
string symbol = Symbol();
// Obter a hora da vela atual
datetime currentTime = iTime(symbol, PeríodoOperacional, 0);
//+-----+
//| Expert deinitialization function |
//+-----+
var1 = compra;
var2 = venda;
pc = var1 ;
pv = var2 ;
Pontos = pc - pv;
divisao = Pontos / 2;
linhabuy = pc + Pontos;

linhasubsiculc = pc + divisao;
//+-----+
//| Expert deinitialization function |
//+-----+
pc = var1 ;
pv = var2 ;
Pontos = pc - pv;
divisao = Pontos / 2;
linhasell = pv - Pontos;
linhasubsiculv = pv - divisao;
//+-----+
//| |
//+-----+
criarlinhavenda("prevodevenda", pv);
criarlinhascompra("precode compra ", pc);
if(ciclos)
{
double primeiro_siclo_de_alta = var1;
double primeiro_siclo_de_baixa = var2;
double segundo_siclo_de_baixa = primeiro_siclo_de_alta - (pontosf * 2);
double terceiro_siclo_de_baixa = primeiro_siclo_de_alta - (pontosf * 4);
double quarto_siclo_de_baixa = primeiro_siclo_de_alta - (pontosf * 8);
double sub_nivel_de_venda = primeiro_siclo_de_baixa - divisao;
double segundo_siclo_de_alta = primeiro_siclo_de_baixa + (pontosf * 2);
double terceiro_siclo_de_alta = primeiro_siclo_de_baixa + (pontosf * 4);
double quarto_siclo_de_alta = primeiro_siclo_de_baixa + (pontosf * 8);
double sub_nivel_de_compra = primeiro_siclo_de_alta + divisao;

// criação de linha de ciclo no grafico
criarlinha_segundo_siclo("siclo2", segundo_siclo_de_alta);
criarlinha_terceiro_siclo("siclo3", terceiro_siclo_de_alta);
criarlinha_quarto_siclo("siclo4", quarto_siclo_de_alta);

criarlinha_2_siclo("siclo2v", segundo_siclo_de_baixa);
criarlinha_3_siclo("siclo3v", terceiro_siclo_de_baixa);
criarlinha_4_siclo("siclo4v", quarto_siclo_de_baixa);

}

return(INIT_SUCCEEDED);
}
//+-----+
//| Expert deinitialization function |
//+-----+
void OnDeinit(const int reason)
{
ObjectsDeleteAll(0, "precocompra", 0, OBJ_HLINE);
ObjectsDeleteAll(0, "precovanda", 0, OBJ_HLINE);
ObjectsDeleteAll(0, "linha3_", 0, OBJ_HLINE);
ObjectsDeleteAll(0, "linha2_", 0, OBJ_HLINE);
ObjectsDeleteAll(0, "linha1_", 0, OBJ_HLINE);
ObjectsDeleteAll(0, "linha_", 0, OBJ_HLINE);
}
//+-----+
//| Expert tick function
//+-----+
void OnTick()
{

```

```

// double nivel = MathAbs(PrecioDeCompra - PrecioDeVenta) / SymbolInfoDouble(_Symbol, SYMBOL_POINT);
double precocurrentecB = SymbolInfoDouble(_Symbol, SYMBOL_BID);
double precocurrentevA = SymbolInfoDouble(_Symbol, SYMBOL_ASK);
double closeprice = iClose(Symbol(), PeriodoOperacional, 1);
double precodiferentev = MathAbs(closeprice - PrecioDeVenta) / Point();
double precodiferentec = MathAbs(PrecioDeCompra - closeprice) / Point();
datetime current_time = TimeLocal();
string symbol = Symbol();
datetime currentTime = iTime(symbol, PeriodoOperacional, 0);
int totalPositions = PositionsTotal();
int bingala_de_alta = int(tamanho_DA_vela_para_compra);
int espasao = int (pontof / Point());
int canais = (bingala_de_alta / espasao);
int bingala_de_baixa = int (tamanho_DA_vela_para_venta);
int canaisv = (bingala_de_baixa / espasao);

if(current_time >= target_time)
{
    // se o tempo for maior que determinado acione algo
}

//+-----+
//| CRIAÇÃO DE LINHAS |
//+-----+
if(closeprice > pc)
{
    criarlinhaBuy1(linhasubsiculc);
    criarlinhaBuy2(linhabuy);
}

//+-----+
//| CRIAÇÃO DE LINHAS |
//+-----+
if(closeprice > linhabuy)
{
    pc = pc + Pontos ;
    pv = pv + Pontos ;
    Pontos = pc - pv;
    divisao = Pontos / 2;
    linhabuy = pc + Pontos;
    linhasubsiculc = pc + divisao;
    linhasell = pv - Pontos;
    linhasubsiculv = pv - divisao;
}

//+-----+
//| CRIAÇÃO DE LINHAS |
//+-----+
if(closeprice < pv)
{
    criarlinhasell1(linhasubsiculv);
    criarlinhasell2(linhasell);
}

//+-----+
//| CRIAÇÃO DE LINHAS |
//+-----+
if(closeprice < linhasubsiculv)
{
    pc = pc - Pontos ;
    pv = pv - Pontos ;
    Pontos = pc - pv;
    divisao = Pontos / 2;
    linhasell = pv - Pontos;
    linhasubsiculv = pv - divisao;
    linhabuy = pc + Pontos;
    linhasubsiculc = pc + divisao;
}

//+-----+
//|
//+-----+

if(precocurrentecB >= takbuy)
{
    if(!segundo_control_de_takbuy)
    {
        PrecioDeVenta = PrecioDeCompra - pontof;
        Buysake = PrecioDeCompra + (pontof * Nives);
        Buystop = PrecioDeCompra - pontof;
        Buysubsicul = PrecioDeCompra + divisao;
        Buymodif = PrecioDeCompra;
        Selltake = PrecioDeVenta - (pontof * Nives);
        Sellestop = PrecioDeVenta + pontof;
        Sellsubsicul = PrecioDeVenta - divisao;
        SellModif = PrecioDeVenta;
    }
}

```

```

        segundo_control_de_takbuy = true;
    }
}

//+-----+
//|
//+-----+

if(pecocurrentevA <= taksell)
{
    if(!segundo_control_de_taksell)
    {
        PrecoDeCompra = PrecoDeVenda + pontosf;
        Buytake = PrecoDeCompra + (pontosf * Nives);
        Buystop = PrecoDeCompra - pontosf;
        Buysubsicul = PrecoDeCompra + divisao;
        Buymodif = PrecoDeCompra;
        Selltake = PrecoDeVenda - (pontosf * Nives);
        Sellestop = PrecoDeVenda + pontosf;
        Sellsubsicul = PrecoDeVenda - divisao;
        SellModif = PrecoDeVenda;

        segundo_control_de_taksell = true;
    }

    //+-----+
    //|
    //+-----+

    if(pecocurrentecB || pecocurrentevA <= nivelstoplos_buy)
    {
        if(totalPositions == 0)
        {
            controlsell = false;
        }
    }
}

if(closeprice > PrecoDeCompra)
{
    if(pecocurrentecB || pecocurrentevA >= nivelstoplos_sell)
    {
        if(totalPositions == 0)
        {
            controlbuy = false;
        }
    }
}

//+-----+
//|
//+-----+

// Obter a última vela
MqlRates rates[1];
// Copiar os dados da última vela para a matriz 'rates'
int copied = CopyRates(symbol, PeríodoOperacional, 0, 1, rates);
if(copied > 0)
{
    // Última vela
    double lastOpen = rates[0].open;
    double lastClose = rates[0].close;
    // Calcular o tamanho da vela
    if(lastOpen > lastClose)
    {
        tamanho_DA_vela_para_compra = MathAbs(lastClose - PrecoDeCompra) / Point();
    }
    if(lastOpen < lastClose)
    {
        tamanho_DA_vela_para_venda = MathAbs(lastClose - PrecoDeVenda) / Point();
    }
}
else
{
    Print("Erro ao copiar os dados da vela: ", GetLastError());
}

//+-----+
//|
//+-----+

if(GERENCIAMENTO_DE_RISCO)
{
    if(pecocurrentecB || pecocurrentevA == PrecoDeCompra || PrecoDeVenda)
    {
        funcao_verifica_meta_ou_perda_atingida();
    }
}
}

```

```
//+-----+
//|
//+-----+
```

```
if(ciclos)
{
    double primeiro_siclo_de_alta = var1;
    double primeiro_siclo_de_baixa = var2;
    double segundo_siclo_de_baixa = primeiro_siclo_de_alta - (pontof * 2);
    double terceiro_siclo_de_baixa = primeiro_siclo_de_alta - (pontof * 4);
    double quarto_siclo_de_baixa = primeiro_siclo_de_alta - (pontof * 8);
    double sub_nivel_de_venda = primeiro_siclo_de_baixa - divisao;
    double segundo_siclo_de_alta = primeiro_siclo_de_baixa + (pontof * 2);
    double terceiro_siclo_de_alta = primeiro_siclo_de_baixa + (pontof * 4);
    double quarto_siclo_de_alta = primeiro_siclo_de_baixa + (pontof * 8);
    double sub_nivel_de_compra = primeiro_siclo_de_alta + divisao;
    double Meiode_de_cliclo3e4_alta = terceiro_siclo_de_alta + (pontof * 2);
    double Meiode_de_cliclo3e4_baixa = terceiro_siclo_de_baixa - (pontof * 2);

    double exp4_altabuy = quarto_siclo_de_alta + (pontof);
    double exp4_baixabuy = quarto_siclo_de_alta - (divisao);
    double exp3e4_DE_altabuy = Meiode_de_cliclo3e4_alta + (pontof);
    double exp3e4_DE_baixabuy = Meiode_de_cliclo3e4_alta - (divisao);
    double exp3altabuy = terceiro_siclo_de_alta + (pontof);
    double exp3baixabuy = terceiro_siclo_de_alta - (divisao);

    double exp4_altasell = quarto_siclo_de_baixa + (divisao);
    double exp4_baixasell = quarto_siclo_de_baixa - (pontof);
    double exp3e4_DE_altasell = Meiode_de_cliclo3e4_baixa + (divisao);
    double exp3e4_DE_baixasell = Meiode_de_cliclo3e4_baixa - (pontof);
    double exp3altasell = terceiro_siclo_de_baixa + (divisao);
    double exp3baixasell = terceiro_siclo_de_baixa - (pontof);

    if(closeprice > primeiro_siclo_de_alta)
    {
        if(ctr21)
        {
            ctr22 = true;
            ctr11 = false;
            ctr1 = true;
            ctr21 = false;
        }
    }

    if(closeprice < primeiro_siclo_de_baixa)
    {
        if(ctr22)
        {
            ctr21 = true;
            ctr11 = true;
            ctr1 = false;
            ctr22 = false;
        }
    }

    if(closeprice > sub_nivel_de_compra)
    {
        if(ctr1)
        {
            control_de_venda = true;
            control_de_venda2 = true;
            Print(" venda feixada ctr1 de venda aberto");
        }

        if(closeprice > exp3baixabuy)
        {
            if(ctr2)
            {
                ctr1 = false;
                control_de_venda = false;
                control_de_venda2 = false;
                Print(" venda aberta ctr1 de venda feixada por exp3baixabuy ");

                ctr3 = true;
                ctr2 = false;
            }

            if(closeprice > exp3altabuy)
            {
                if(ctr3)
                {
```

```

        Print(" venda feixada ctr1 de venda feixada por exp3altabuy ");
        ctr4 = true;
        ctr1 = true;
        ctr3 = false;
    }

}

}
if(closeprice < segundo_siclo_de_alta && ctr3 == false)
{
    if(ctr4)
    {
        ctr3 = true;
        ctr1 = false;
        control_de_venda = false;
        control_de_venda2 = false;
        Print(" venda aberta ctr1 de venda aberta por segunda ciclo de alta ");

        ctr4 = false;
    }

}

if(closeprice > exp3e4_DE_baixabuy) /// abrir venda
{
    if(ctr5)
    {
        ctr1 = false;
        Print(" venda aberta ctr1 de venda aberto maior baixa que 3e4 by");

        control_de_venda = false;
        control_de_venda2 = false;
        ctr5 = false;
    }
    if(closeprice > exp3e4_DE_altabuy) // fechar venda
    {
        if(ctr6)
        {
            ctr1 = true;
            Print(" venda feixada ctr1 de venda aberto mair que alta 3e4 buy");
            ctr7 = true;
            control_de_venda = true;
            control_de_venda2 = true;
            ctr6 = false;

        }
    }
}

if(ctr6 == false && closeprice < terceiro_siclo_de_alta) // abrir venda se for menor qure terceiro nivel
{
    if(ctr7)
    {
        ctr6 = true;
        ctr1 = false;
        Print(" venda aberta ctr1 de venda feixado se ctr6 == false");

        control_de_venda = false;
        control_de_venda2 = false;
        ctr7 = false;
    }

}

if(closeprice > exp4_baixabuy) /// abrir a venda
{
    if(ctr8)
    {
        Print(" venda aberta ctr1 de venda feicado por exp4 baixa");

        ctr1 = false;
        control_de_venda = false;
        control_de_venda2 = false;

        ctr8 = false;
    }
    if(closeprice > exp4_altabuy) // fechar a venda

```



```

    {
        if(ctr9)
        {
            Print(" venda feixada ctr1 de venda aberta por exp4 alta");
            ctr10 = true;
            ctr1 = true;
            control_de_venda = true;
            control_de_venda2 = true;
            ctr9 = false;
        }

    }

}

if(ctr9 == false && closeprice < Meiode_de_cliclo3e4_alta)
// abrir a venda se o feixamento for menor que meio de 3e4
{
    if(ctr10)
    {
        Print(" venda aberta ctr1 de venda feicado se ctr9 == false meio de 3e4 alta");
        ctr9 = true;
        ctr1 = false;
        control_de_venda2 = false;
        control_de_venda = false;
        ctr10 = false;
    }
}

}

//+-----+
//|          surfadas de venda por ciclos          |
//+-----+

if(closeprice < sub_nivel_de_venda) // faixar compra
{
    if(ctr11)
    {
        Print(" compra feixada ctr11 de compra aberto por sub venda");

        control_de_compra = true;
        control_de_compra2 = true;

    }

if(closeprice < exp3altasell) // abri compra
{
    if(ctr12)
    {
        Print(" compra aberto ctr11 de compra feixado por exp3 alta sell");

        ctr11 = false;
        control_de_compra = false;
        control_de_compra2 = false;

        ctr12 = false;

    }
}

if(closeprice < exp3baixasell) // feixar compra
{
    if(ctr13)
    {
        Print(" compra feixada ctr11 de compra aberto por exp3 baixa sell");
        ctr14 = true;
        ctr11 = true;
        ctr13 = false;
    }

}

if(closeprice > segundo_siclo_de_baixa && ctr13 == false)
{
    if(ctr14)

```

```

{
    Print(" compra abetrta ctr11 de compra feixado se ctr13 == false por segundo ciclo de baixa ");
    ctr13 = true;
    ctr11 = false;
    control_de_compra = false;
    control_de_compra2 = false;
    ctr14 = false;
}

}

if(closeprice < exp3e4_DE_altasell) /// abrir compra
{
    if(ctr15)
    {
        Print(" compra aberto ctr11 de compra feixada por esp3e4 alta sell", exp3e4_DE_baixasell);

        ctr11 = false;
        control_de_compra = false;
        control_de_compra2 = false;
        ctr15 = false;
    }
}

if(closeprice < exp3e4_DE_baixasell) // fechar compra
{
    if(ctr16)
    {
        Print(" compra feixada ctr11 de compra aberto por exp3e4 baixa sell", exp4_altasell);
        ctr17 = true;
        ctr11 = true;
        control_de_compra = true;
        control_de_compra2 = true;
        ctr16 = false;
    }

}

if(ctr16 == false && closeprice > terceiro_siclo_de_baixa) // abrir compra se for maior que terceiro nivel
{
    if(ctr17)
    {
        Print(" compra feixada ctr11 de compra aberto se ctr16 == false por terceira baixa ");
        ctr16 = true;
        ctr11 = false;
        control_de_compra = false;
        control_de_compra2 = false;
        ctr17 = false;
    }

}

if(closeprice < exp4_altasell) // abrir a compra
{
    if(ctr18)
    {
        Print(" compra aberto ctr11 de compra feixada por exp4 alta sell");

        ctr11 = false;
        control_de_compra = false;
        control_de_compra2 = false;

        ctr18 = false;
    }
}

if(closeprice < exp4_baixasell) // fechar a compra
{
    if(ctr19)
    {
        Print(" compra feixada ctr11 de compra aberto por exp4 baixa sell");
        ctr20 = true;
        ctr11 = true;
        control_de_compra = true;
        control_de_compra2 = true;
        ctr19 = false;
    }
}

```

```

    }
    if(closeprice > Meiode_de_cliclo3e4_baixa && ctr19 == false)
// abrir a venda se o feixamento for maior que meio de 3e4
    {
        if(ctr20)
        {
            Print(" compra aberta ctr11 de compra feixado se ctr 19 == falso meio de 3e4", Meiode_de_cliclo3e4_baixa);
            ctr19 = true;
            ctr11 = false;
            control_de_compra = false;
            control_de_compra2 = false;
            ctr20 = false;
        }
    }

}

//+-----+
//|
//+-----+

}
//+-----+
//|
//+-----+

//+-----+
//|
//+-----+

//+-----+
//|
//+-----+

if(surfada)
{
    if(!controlbuymdf)
    {
        if(closeprice > Buysubsicul)
        {
            if(totalPositions == 1)
            {
                if(bingala_de_alta > espasao)
                {
                    for(int i = Nives ; i <= canais ; i++)
                    {
                        PrecoDeVenda = PrecoDeCompra;
                        PrecoDeCompra = Buytake;
                        Buytake = PrecoDeCompra + (pontofsf * Nives);
                        Buystop = PrecoDeCompra - pontofsf;
                        Buysubsicul = PrecoDeCompra + divisao;
                        Buymodif = PrecoDeVenda;
                        Selltake = PrecoDeVenda - (pontofsf * Nives) ;
                        Sellestop = PrecoDeVenda + pontofsf;
                        Sellsubsicul = PrecoDeVenda - divisao;
                        SellModif = PrecoDeVenda;
                    }
                    ModifyBuyOrder();
                }
            }
            else
            {
                ModifyBuyOrder();
                PrecoDeVenda = PrecoDeCompra;
                PrecoDeCompra = Buytake;
                Buytake = PrecoDeCompra + (pontofsf * Nives);
                Buystop = PrecoDeCompra - pontofsf;
                Buysubsicul = PrecoDeCompra + divisao;
                Buymodif = PrecoDeCompra;
                Selltake = PrecoDeVenda - (pontofsf * Nives) ;
                Sellestop = PrecoDeVenda + pontofsf;
                Sellsubsicul = PrecoDeVenda - divisao;
                SellModif = PrecoDeVenda;
            }
        }
    }
}

```

```

    }

    if(Costura)
    {
        control_de_venda = false;
        control_de_compra = false;
    }

    controlbuy = false;
    controlsell = false;
    controlsellmdf = false;
    controlbuymdf = true;
}
else
{
    if(bingala_de_alta > espasao)
    {
        for(int i = Nives ; i <= canais ; i++)
        {
            PrecoDeVenda = PrecoDeCompra;
            PrecoDeCompra = Buytake;
            Buytake = PrecoDeCompra + (pontofsf * Nives);
            Buystop = PrecoDeCompra - pontofsf;
            Buysubsicul = PrecoDeCompra + divisao;
            Buymodif = PrecoDeCompra;
            Selltake = PrecoDeVenda - (pontofsf * Nives) ;
            Sellestop = PrecoDeVenda + pontofsf;
            Sellsubsicul = PrecoDeVenda - divisao;
            SellModif = PrecoDeVenda;

        }
    }

    else
    {
        PrecoDeVenda = PrecoDeCompra;
        PrecoDeCompra = Buytake;
        Buytake = PrecoDeCompra + (pontofsf * Nives) ;
        Buystop = PrecoDeCompra - pontofsf;
        Buysubsicul = PrecoDeCompra + divisao;
        Buymodif = PrecoDeCompra;
        Selltake = PrecoDeVenda - (pontofsf * Nives) ;
        Sellestop = PrecoDeVenda + pontofsf;
        Sellsubsicul = PrecoDeVenda - divisao;
        SellModif = PrecoDeVenda;

    }

    if(Costura)
    {
        control_de_venda = false;
        control_de_compra = false;
    }

    controlbuy = false;
    controlsell = false;
    controlsellmdf = false;
    controlbuymdf = true;
}
}
}
}
else
{
    if(!controlbuymdf)
    {
        if(closeprice > Buysubsicul)
        {
            if(bingala_de_alta > espasao)
            {
                for(int i = Nives ; i <= canais ; i++)
                {
                    PrecoDeVenda = PrecoDeCompra;
                    PrecoDeCompra = Buytake;
                    Buytake = PrecoDeCompra + (pontofsf * Nives);
                    Buystop = PrecoDeCompra - pontofsf;
                    Buysubsicul = PrecoDeCompra + divisao;
                    Buymodif = PrecoDeVenda;
                    Selltake = PrecoDeVenda - (pontofsf * Nives) ;
                    Sellestop = PrecoDeVenda + pontofsf;
                    Sellsubsicul = PrecoDeVenda - divisao;
                    SellModif = PrecoDeVenda;

                }
            }
        }
    }
}

```

```

        ModifyBuyOrder();
    }
else
{
    ModifyBuyOrder();
    PrecoDeVenda = PrecoDeCompra;
    PrecoDeCompra = Buytake;
    Buytake = PrecoDeCompra + (pontof * Nives);
    Buystop = PrecoDeCompra - pontof;
    Buysubsicul = PrecoDeCompra + divisao;
    Buymodif = PrecoDeCompra;
    Selltake = PrecoDeVenda - (pontof * Nives);
    Sellestop = PrecoDeVenda + pontof;
    Sellsubsicul = PrecoDeVenda - divisao;
    SellModif = PrecoDeVenda;

}

if(Costura)
{
    control_de_compra2 = false;
    control_de_venda2 = false;
}
controlsellmdf = false;
controlsell = false;
controlbuy = false;
controlbuymdf = true;
}
}

//+-----+
//|
//+-----+

//+-----+
//|
//+-----+

bool condicao2 = (condicao_De_rompimento_c) ? // condição para rompimento da vela de compra
    (precodiferentec < dedo) : true ;

if(surfada)
{
    if(closeprice > PrecoDeCompra)
    {
        if(! controlbuy)
        {
            if(totalPositions == 1)
            {
                PrecoDeCompra = PrecoDeCompra ;
                PrecoDeVenda = PrecoDeCompra - pontof ;
                Buytake = PrecoDeCompra + (pontof * Nives);
                Buystop = PrecoDeCompra - pontof;
                Buysubsicul = PrecoDeCompra + divisao;
                Buymodif = PrecoDeCompra;
                Selltake = PrecoDeVenda - (pontof * Nives);
                Sellestop = PrecoDeVenda + pontof;
                Sellsubsicul = PrecoDeVenda - divisao;
                SellModif = PrecoDeVenda;

                controlsell = false ;
                controlbuymdf = false;
                controlsellmdf = false;
                segundo_control_de_takbuy = false;
                segundo_control_de_taksell = false;
                controlbuy = true;
            }
        }
    }
else
{
    if(! control_de_compra)
    {
        if(ativar_ou_desativar_compra && condicao2)
        {
            BuyMarkt();
        }
        PrecoDeCompra = PrecoDeCompra ;
        PrecoDeVenda = PrecoDeCompra - pontof ;
        Buytake = PrecoDeCompra + (pontof * Nives) ;
        Buystop = PrecoDeCompra - pontof;
        Buysubsicul = PrecoDeCompra + divisao;
        Buymodif = PrecoDeCompra;
        Selltake = PrecoDeVenda - (pontof * Nives);
    }
}
}

```

```

        Sellestop = PrecoDeVenda + pontosf;
        Sellsubsicul = PrecoDeVenda - divisao;
        SellModif = PrecoDeVenda;
        control_de_compra = true;
    }

    controlsell = false ;
    controlbuy MDF = false;
    controlsell MDF = false;
    segundo_control_de_takbuy = false;
    segundo_control_de_taksell = false;
    controlbuy = false;
}
}
}
}
else
{
    if(!controlbuy)
    {
        if(closeprice > PrecoDeCompra)
        {
            if(!control_de_compra2)
            {
                if(totalPositions == 0)
                {
                    if(ativar_ou_desativar_compra && condicao2)
                    {
                        BuyMarkt();
                        control_de_compra2 = true;
                    }
                }
            }
            PrecoDeVenda = PrecoDeCompra - pontosf ;
            Buytake = PrecoDeCompra + (pontosf * Nives);
            Buystop = PrecoDeCompra - pontosf;
            Buysubsicul = PrecoDeCompra + divisao;
            Buymodif = PrecoDeCompra;
            Selltake = PrecoDeVenda - (pontosf * Nives);
            Sellestop = PrecoDeVenda + pontosf;
            Sellsubsicul = PrecoDeVenda - divisao;
            SellModif = PrecoDeVenda;

            controlsell = false ;
            controlbuy MDF = false;
            controlsell MDF = false;
            segundo_control_de_takbuy = false;
            segundo_control_de_taksell = false;
            controlbuy = true;
        }
    }
}

//+-----+
//|
//+-----+

//+-----+
//|
//+-----+

if(surfada)
{
    if(! controlsell MDF)
    {
        if(closeprice < Sellsubsicul)
        {
            if(totalPositions == 1)
            {
                if(bingala_de_baixa > espasao)
                {
                    for(int i = Nives ; i <= canaisv ; i++)
                    {
                        PrecoDeCompra = PrecoDeVenda;
                        PrecoDeVenda = Selltake;
                        Buytake = PrecoDeCompra + (pontosf * Nives) ;
                        Buystop = PrecoDeCompra - pontosf;
                        Buysubsicul = PrecoDeCompra + divisao;
                        Buymodif = PrecoDeCompra;
                        Selltake = PrecoDeVenda - (pontosf * Nives) ;
                        Sellestop = PrecoDeVenda + pontosf;
                        Sellsubsicul = PrecoDeVenda - divisao;
                        SellModif = PrecoDeCompra;
                    }
                }
            }
        }
    }
}

```

```

        ModifySellOrder();
    }
else
{

    ModifySellOrder();
    PrecoDeCompra = PrecoDeVenda;
    PrecoDeVenda = Selltake;
    Buytake = PrecoDeCompra + (pontofsf * Nives) ;
    Buystop = PrecoDeCompra - pontofsf;
    Buysubsicul = PrecoDeCompra + divisao;
    Buymodif = PrecoDeCompra;
    Selltake = PrecoDeVenda - (pontofsf * Nives) ;
    Sellestop = PrecoDeVenda + pontofsf;
    Sellsubsicul = PrecoDeVenda - divisao;
    SellModif = PrecoDeVenda;

}

if(Costura)
{
    control_de_compra = false;
    control_de_venda = false;
}
controlbuy = false;
controlsell = false;
controlbuymdf = false;
controlsellmdf = true;
}
else
{
    if(bingala_de_baixa > espasao)
    {
        for(int i = Nives ; i <= canaisv ; i++)
        {
            PrecoDeCompra = PrecoDeVenda;
            PrecoDeVenda = Selltake;
            Buytake = PrecoDeCompra + (pontofsf * Nives);
            Buystop = PrecoDeCompra - pontofsf;
            Buysubsicul = PrecoDeCompra + divisao;
            Buymodif = PrecoDeCompra;
            Selltake = PrecoDeVenda - (pontofsf * Nives);
            Sellestop = PrecoDeVenda + pontofsf;
            Sellsubsicul = PrecoDeVenda - divisao;
            SellModif = PrecoDeVenda;

        }
    }
else
{
    PrecoDeCompra = PrecoDeVenda;
    PrecoDeVenda = Selltake;
    Buytake = PrecoDeCompra + (pontofsf * Nives) ;
    Buystop = PrecoDeCompra - pontofsf;
    Buysubsicul = PrecoDeCompra + divisao;
    Buymodif = PrecoDeCompra;
    Selltake = PrecoDeVenda - (pontofsf * Nives) ;
    Sellestop = PrecoDeVenda + pontofsf;
    Sellsubsicul = PrecoDeVenda - divisao;
    SellModif = PrecoDeVenda;

}

if(Costura)
{
    control_de_compra = false;
    control_de_venda = false;
}
controlbuy = false;
controlsell = false;
controlbuymdf = false;
controlsellmdf = true;
}
}
}
}
else
{
    if(!controlsellmdf)
    {
        if(closeprice < Sellsubsicul)

```

```

{
    if(bingala_de_baixa > espassao)
    {
        for(int i = Nives ; i <= canaisv ; i++)
        {
            PrecoDeCompra = PrecoDeVenda;
            PrecoDeVenda = Selltake;
            Buytake = PrecoDeCompra + (pontof * Nives);
            Buystop = PrecoDeCompra - pontof;
            Buysubsicul = PrecoDeCompra + divisao;
            Buymodif = PrecoDeCompra;
            Selltake = PrecoDeVenda - (pontof * Nives) ;
            Sellestop = PrecoDeVenda + pontof;
            Sellsuscul = PrecoDeVenda - divisao;
            SellModif = PrecoDeCompra;

        }

        ModifySellOrder();
    }

    else
    {
        ModifySellOrder();
        PrecoDeCompra = PrecoDeVenda;
        PrecoDeVenda = Selltake;
        Buytake = PrecoDeCompra + (pontof * Nives);
        Buystop = PrecoDeCompra - pontof;
        Buysubsicul = PrecoDeCompra + divisao;
        Buymodif = PrecoDeCompra;
        Selltake = PrecoDeVenda - (pontof * Nives);
        Sellestop = PrecoDeVenda + pontof;
        Sellsuscul = PrecoDeVenda - divisao;
        SellModif = PrecoDeVenda;

    }

    if(Costura)
    {
        control_de_venda2 = false;
        control_de_compra2 = false;
    }
    controlbuymdf = false;
    controlbuy = false;
    controlsell = false;
    controlsellmdf = true;
}
}

}

//+-----+
//|
//+-----+

//+-----+
//|
//+-----+

bool condicao = (condicao_De_rompimento_v) ?
                (precodiferentev < dedo) : true ;

if(surfada)
{
    if(closeprice < PrecoDeVenda)
    {
        if(!controlsell)
        {
            if(totalPositions == 1)
            {

                PrecoDeVenda = PrecoDeVenda ;
                PrecoDeCompra = PrecoDeVenda + pontof;
                Buytake = PrecoDeCompra + (pontof * Nives) ;
                Buystop = PrecoDeCompra - pontof;
                Buysubsicul = PrecoDeCompra + divisao;
                Buymodif = PrecoDeCompra;
                Selltake = PrecoDeVenda - (pontof * Nives) ;
                Sellestop = PrecoDeVenda + pontof;
                Sellsuscul = PrecoDeVenda - divisao;
                SellModif = PrecoDeVenda;

                controlbuy = false;
                controlsellmdf = false;
                controlbuymdf = false;
                segundo_control_de_takbuy = false;
                segundo_control_de_taksell = false;
            }
        }
    }
}

```



```

//+-----+
//|
//+-----+
bool BuyMarkt()
{
    MqlTradeRequest request = { };
    MqlTradeResult result = { };
    request.action = TRADE_ACTION_DEAL;
    request.volume = lot;
    request.symbol = Symbol();
    request.type = ORDER_TYPE_BUY;
    request.magic = MagicNumber;
    if(surfada)
    {
        request.sl = Buystop - fora ;
    }
    else
    {
        request.sl = Buystop - fora ;
        request.tp = Buytake - fora ;
    }
    takbuy = request.tp;
    nivelstoplos_buy = request.sl;
    if(OrderSend(request, result))
    {
        BilheteDeCompra = result.order;
        preco_de_abertura_de_compra = result.price;
        Print("ordem de compra enviada -Bilhete : ", BilheteDeCompra);
        return true;
    }
    else
    {
        Print("Erro ao enviar a ordem de compra -erro :", GetLastError());
        return false;
    }
}
//+-----+
bool SellMarkt()
{
    MqlTradeRequest request = { };
    MqlTradeResult result = { };
    request.action = TRADE_ACTION_DEAL;
    request.volume = lot;
    request.symbol = Symbol();
    request.type = ORDER_TYPE_SELL;
    request.magic = MagicNumber;

    if(surfada)
    {
        request.sl = Sellestop + fora ;
    }
    else
    {
        request.sl = Sellestop + fora ;
        request.tp = Selltake + fora ;
    }
    taksell = request.tp;
    nivelstoplos_sell = request.sl;
    if(OrderSend(request, result))
    {
        BilheteDeVenda = result.order;
        preco_de_abertura_de_venda = result.price;
        Print("ordem de venda enviada -Bilhete : ", BilheteDeVenda);
        return true;
    }
    else
    {
        Print("Erro ao enviar a ordem de venda -erro :", GetLastError());
        return false;
    }
    return true;
}
//+-----+
bool ModifySellOrder()
{
    if(PositionSelectByTicket(BilheteDeVenda))
    {
        {
            MqlTradeRequest request = { };
            MqlTradeResult result = { };
            request.action = TRADE_ACTION_SLTP;
            request.symbol = Symbol();
            request.magic = MagicNumber;

```

```

        if(surfada)
        {
            request.sl = SellModif ;

        }
    else
    {
        request.tp = Selltake + fora;
        if(Modificar_SL_Para_Ox0)
        {
            request.sl = preco_de_abertura_de_venda ;
        }
        else
        {
            request.sl = SellModif ;
        }
    }
    request.position = BilheteDeVenda;
    nivelzerosell = request.sl;
    taksell = request.tp;
    if(OrderSend(request, result))
    {
        Print("Ordem de venda modificada - Bilhete: ", BilheteDeVenda, " Novo SL: ", SellModif);
        return true;
    }
    else
    {
        Print("Erro ao modificar a ordem de venda - Bilhete: ", BilheteDeVenda);
        return false;
    }
}

return true;
}
}
//+-----+
//| |
//+-----+
bool ModifyBuyOrder()
{
    if(PositionSelectByTicket(BilheteDeCompra))
    {
        {
            MqlTradeRequest request = { };
            MqlTradeResult result = { };
            request.action = TRADE_ACTION_SLTP;
            request.symbol = Symbol();
            request.magic = MagicNumber;

            if(surfada)
            {
                request.sl = Buymodif;

            }
            else
            {
                request.tp = Buytake - fora;
                if(Modificar_SL_Para_Ox0)
                {
                    request.sl = preco_de_abertura_de_compra;
                }
                else
                {
                    request.sl = Buymodif;
                }
            }
            request.position = BilheteDeCompra;
            nivelzerobuy = request.sl;
            takbuy = request.tp;
            if(OrderSend(request, result))
            {
                Print("Ordem de compra modificada - Bilhete: ", BilheteDeCompra, " Novo SL: ", Buymodif);
                return true;
            }
            else
            {
                Print("Erro ao modificar a ordem de compra - Bilhete: ", BilheteDeCompra);
            }
            return false;
        }
    }
}

return true;
}

```

```

    }
//+-----+
void criarlinhassell1(double valor)
{
    string subsicul = "linha3_" + DoubleToString(valor, 4);
    bool unica3 = true;
    for(int i = 0; i < ArraySize(precosArray); i++)
    {
        if(precosArray[i] == valor)
        {
            unica3 = false ;
            break;
        }
    }
    if(unica3)
    {
        int currentSize = ArraySize(precosArray);
        ArrayResize(precosArray, currentSize + 1);
        precosArray[currentSize] = valor;
        string subsicul = "linha3_" + DoubleToString(valor, 4);
        ObjectCreate(0, subsicul, OBJ_HLINE, 0, 0, valor);
        ObjectSetInteger(0, subsicul, OBJPROP_STYLE, STYLE_SOLID);
        ObjectSetInteger(0, subsicul, OBJPROP_COLOR, clrBlue);
    }
}
//+-----+
void criarlinhassell2(double valor)
{
    string precosell = "linha2_" + DoubleToString(valor, 4);
    bool unica2 = true;
    for(int i = 0; i < ArraySize(precosArray); i++)
    {
        if(precosArray[i] == valor)
        {
            unica2 = false ;
            break;
        }
    }
    if(unica2)
    {
        int currentSize = ArraySize(precosArray);
        ArrayResize(precosArray, currentSize + 1);
        precosArray[currentSize] = valor;
        string precosell = "linha2_" + DoubleToString(valor, 4);
        ObjectCreate(0, precosell, OBJ_HLINE, 0, 0, valor);
        ObjectSetInteger(0, precosell, OBJPROP_STYLE, STYLE_SOLID);
        ObjectSetInteger(0, precosell, OBJPROP_COLOR, clrRed);
    }
}
//+-----+
//| // criar linha de compra vermelho |
//+-----+
void criarlinhaBuy1(double valor)
{
    string subsiculbuy = "linha1_" + DoubleToString(valor, 4);
    bool unica1 = true;
    for(int i = 0; i < ArraySize(precosArray); i++)
    {
        if(precosArray[i] == valor)
        {
            unica1 = false ;
            break;
        }
    }
    if(unica1)
    {
        int currentSize = ArraySize(precosArray);
        ArrayResize(precosArray, currentSize + 1);
        precosArray[currentSize] = valor;
        string subsiculbuy = "linha1_" + DoubleToString(valor, 4);
        ObjectCreate(0, subsiculbuy, OBJ_HLINE, 0, 0, valor);
        ObjectSetInteger(0, subsiculbuy, OBJPROP_STYLE, STYLE_SOLID);
        ObjectSetInteger(0, subsiculbuy, OBJPROP_COLOR, clrBlue);
    }
}
//+-----+
void criarlinhaBuy2(double valor)
{
    string precobuy = "linha_" + DoubleToString(valor, 4);
    bool unica = true;
    for(int i = 0; i < ArraySize(precosArray); i++)
    {
        if(precosArray[i] == valor)
        {

```

```

        unica = false ;
        break;
    }
}
if(unica)
{
    int currentSize = ArraySize(precosArray);
    ArrayResize(precosArray, currentSize + 1);
    precosArray[currentSize] = valor;
    string precobuy = "linha_" + DoubleToString(valor, 4);
    ObjectCreate(0, precobuy, OBJ_HLINE, 0, 0, valor);
    ObjectSetInteger(0, precobuy, OBJPROP_STYLE, STYLE_SOLID);
    ObjectSetInteger(0, precobuy, OBJPROP_COLOR, clrRed);
}
}

//+-----+
//|
//+-----+
bool funcao_verifica_meta_ou_perda_atingida()
{
    double tmpValor_Maximo_Ganho ; //tmpValor_Maximo_Ganho = valor estipulado de meta do dia
    double tmpValorMaximoPerda ; //tmpValorMaximoPerda = valor máximo desejado como perda máxima
    double tmp_resultado_financeiro_dia = 0.0;
    string tmp_x;
    int tmp_contador;
    // obter saldo disponível na conta
    double saldototal = AccountInfoDouble(ACCOUNT_MARGIN_FREE);

    // calculos de porcentagem para preda
    tmpValorMaximoPerda = saldototal * porcentos / -100.0;
    tmpValor_Maximo_Ganho = saldototal * poercentosg / 100.0;

    MqlDateTime tmp_data_b;

    TimeCurrent(tmp_data_b);
    tmp_resultado_financeiro_dia = 0;
    tmp_x = string(tmp_data_b.year) + "." + string(tmp_data_b.mon) + "." + string(tmp_data_b.day) + " 00:00:01";

    HistorySelect(StringToTime(tmp_x), TimeCurrent());
    int tmp_total = HistoryDealsTotal();
    ulong tmp_ticket = 0;
    double tmp_price;
    double tmp_profit;
    datetime tmp_time;
    string tmp_symbol1;
    long tmp_typee;
    long tmp_entry;

    //--- para todos os negócios
    for(tmp_contador = 0; tmp_contador < tmp_total; tmp_contador++)
    {
        //--- tentar obter ticket negócios
        if((tmp_ticket = HistoryDealGetTicket(tmp_contador)) > 0)
        {
            //--- obter as propriedades negócios
            tmp_price = HistoryDealGetDouble(tmp_ticket, DEAL_PRICE);
            tmp_time = (datetime)HistoryDealGetInteger(tmp_ticket, DEAL_TIME);
            tmp_symbol1 = HistoryDealGetString(tmp_ticket, DEAL_SYMBOL);
            tmp_typee = HistoryDealGetInteger(tmp_ticket, DEAL_TYPE);
            tmp_entry = HistoryDealGetInteger(tmp_ticket, DEAL_ENTRY);
            tmp_profit = HistoryDealGetDouble(tmp_ticket, DEAL_PROFIT);
            //--- apenas para o símbolo atual
            if(tmp_symbol1 == Symbol())
                tmp_resultado_financeiro_dia = tmp_resultado_financeiro_dia + tmp_profit;
        }
    }

    if(tmp_resultado_financeiro_dia == 0)
    {
        if(tmp_placar == true)
            Comment("Placar 0x0");
        return(false); //sem ordens no dia
    }
    else
    {
        if((tmp_resultado_financeiro_dia > 0) && (tmp_resultado_financeiro_dia != 0))
        {
            if(tmp_placar == true)
                Comment("Lucro R$" + DoubleToString(NormalizeDouble(tmp_resultado_financeiro_dia, 2), 2));

            if(GERENCIAMENTO_DE_RISCO)
            {

```

```

        if(tmp_resultado_financeiro_dia > tmpValor_Maximo_Ganho)
        {

            control_de_compra = true;
            control_de_compra2 = true;
            control_de_venda = true;
            control_de_venda2 = true;

            return(true);
        }
    }
else
{
    if(tmp_placar == true)
        Comment("Prejuizo R$" + DoubleToString(NormalizeDouble(tmp_resultado_financeiro_dia, 2), 2));
    if(GERENCIAMENTO_DE_RISCO)
    {
        if(tmp_resultado_financeiro_dia < tmpValorMaximoPerda)

            control_de_compra = true;
            control_de_compra2 = true;
            control_de_venda = true;
            control_de_venda2 = true;
            return(true);
        }
    }
}
return(false);
}

```

```

//+-----+
//|
//+-----+

```

```

//+-----+
//|
//+-----+

```