**ส่วนโค้ดโปรแกรมไฟล์ logic.mqh**

double TP;

double SL;

double Lots;

double Magic = 111112;

string ea\_name = "da63cc133f012a28d60518a\_MA\_Cross\_"; // กำหนดชื่อ EA

bool Open\_Buy(){ // ฟังค์ชั่น (Function) รับเงื่อนไขคำสั่งซื้อจากผู้ใช้

if(iMA(NULL, PERIOD\_M15, 20, 0, MODE\_EMA, PRICE\_CLOSE, 1) >

iMA(NULL, PERIOD\_M15, 100, 0, MODE\_EMA, PRICE\_CLOSE, 1) &&

iClose(NULL, PERIOD\_M15, 1) >

iMA(NULL,PERIOD\_M15, 20, 0, MODE\_EMA, PRICE\_CLOSE,1)){

return 1;

}else

return 0;

} //end function

bool Open\_Sell(){ // ฟังค์ชั่น (Function) รับเงื่อนไขคำสั่งขายจากผู้ใช้

return 0;

} //end function

bool Close\_Buy(){ // ฟังค์ชั่น (Function) รับเงื่อนไขปิดคำสั่งซื้อจากผู้ใช้

if(iMA(NULL, PERIOD\_M15, 20, 0, MODE\_EMA, PRICE\_CLOSE, 1) <

iMA(NULL, PERIOD\_M1, 100, 0, MODE\_EMA, PRICE\_CLOSE, 1) ){

return 1;

}else

return 0;

} //end function

bool Close\_Sell(){ // ฟังค์ชั่น (Function) รับเงื่อนไขปิดคำสั่งขายจากผู้ใช้

return 0;

} //end function

bool Option(){ //ฟังค์ชั่น (Function) กำหนดเงื่อนไขอื่นๆจากผู้ใช้

TP = 450;

SL = 300;

Lots = 0.01;

if( TP != 0 ){

return 1;

}else

return 0;

} **//**end function

**ส่วนของโค้ดโปรแกรมไฟล์ MM.mq4**

#include <logic.mqh> // ประกาศไฟล์เพื่อดึงเงื่อนไขผู้ใช้จากไฟร์ logic.mql

input bool ForwardTest = true;

int Layer = 0;

short Buy = 0;

short Sell = 0;

double total = 0;

double Buffer = 0;

double Arr\_Loss[5] = {0,0,0,0,0};

double Money = 0;

double stoploss = 0;

double takeprofit = 0;

string Status = "";

double CheckOrders = 0;

short Stat\_Close\_Pos = 0;

double NumWin = 0;

bool handle;

string Arr\_Date[3];

int days = 0; //check days BlackTest

string Days;

string str;

double LastMoney = 0;

int LastPosition = 0;

string comment;

string hour = "";

//================================================================

#define INTERNET\_SERVICE\_FTP 1

#define INTERNET\_SERVICE\_GOPHER 2

#define INTERNET\_SERVICE\_HTTP 3

#define FTP\_TRANSFER\_TYPE\_UNKNOWN 0x00000000

#define FTP\_TRANSFER\_TYPE\_ASCII 0x00000001

#define FTP\_TRANSFER\_TYPE\_BINARY 0x00000002

#import "wininet.dll"

int InternetOpenW(

string Agent,

int AccessType,

string ProxyName,

string ProxyBypass,

int Flags);

int InternetConnectW(

int hInternetSession,

string ServerName,

int ServerPort,

string UserName,

string Password,

int Service,

int Flags,

int Context);

bool FtpPutFileW(

int hFtpSession,

string LocalFile,

string RemoteFile,

int Flags,

int Context);

bool FtpGetFileW(

int hFtpSession,

string lpszRemoteFile,

string lpszNewFile,

bool fFailIfExists,

int dwLocalFlagsAttribute,

int dwInternetFlags,

int dwContext);

bool InternetCloseHandle(

int hInet);

//================================================================

void DownLoadUploadFile(){{ // ฟังค์ชั่น (Function) ทำการส่งไฟล์ขึ้นไปที่เว็บไซต์

if(ForwardTest == true ){ // เงื่อนไขตรวจสอบว่าผู้ใช้ทำการทดสอบจริง

int hIntObj, hIntObjConn;

string Password, ServerName, UserName;

bool success = false;

hIntObj = InternetOpenW(

"MyInternetObject", 1, NULL, NULL, 0);

if(hIntObj>0){

ServerName ="gator4148.hostgator.com"; // ชื่อเซิฟเวอร์ของเรา

UserName = "vertical@doujin69-th.com"; // ID ใช้ส่งไฟล์ขึ้นเซิฟเวอร์

Password = "y4nlfwtkCz#H"; // รหัสผ่านของ ID

hIntObjConn = InternetConnectW(

hIntObj, ServerName, 21, UserName, Password,

INTERNET\_SERVICE\_FTP, 0, 0);

if(hIntObjConn > 0){ //เงื่อนไขตรวจสอบการเชื่อมต่ออินเทอร์เน็ตสำเร็จ

printf("Successfully connected. %d",

hIntObjConn);

Check\_Time();

FileDatetime();

if(Days != Arr\_Date[2]){ // ตรวจสอบมีการเขียนไฟล์ขึ้นหรือยัง

handle = FileOpen(ea\_name + str + ".csv",

FILE\_CSV | FILE\_READ | FILE\_WRITE, ',');

printf("Day %s != Arr\_Days %s",

Days, Arr\_Date[2]);

FileWrite(handle, "Time", "Money",

"Balance", "Position", "Buy", "Sell",

"StatWin", "Buffer", "Layer1", "Layer2",

"Layer3", "Layer4", "\n" +

TimeToStr(TimeLocal(), TIME\_SECONDS),

DoubleToStr(Money, 2),

DoubleToStr(AccountBalance(), 2),

DoubleToStr(total, 0),

DoubleToStr(Buy, 0),

DoubleToStr(Sell, 0),

DoubleToStr(NumWin, 0),

DoubleToStr(Buffer, 2),

DoubleToStr(Arr\_Loss[1], 2),

DoubleToStr(Arr\_Loss[2], 2),

DoubleToStr(Arr\_Loss[3], 2),

DoubleToStr(Arr\_Loss[4], 2));

FileClose(handle); // ปิดไฟล์ที่เขียนขึ้นมา

string LocalFile = TerminalInfoString(

TERMINAL\_DATA\_PATH) "\\MQL4\\Files\\" +

ea\_name + str + ".csv"; // กำหนดที่อยู่ไฟล์ที่เขียนขึ้น

string HostFile = ea\_name + str + ".csv";

success = FtpPutFileW(hIntObjConn, LocalFile,

HostFile, FTP\_TRANSFER\_TYPE\_BINARY, NULL);

// เก็บค่าการส่งไฟล์ขึ้นสู่เว็บไซต์

if(Minute() == 0){ // เงื่อนไขตรวจสอบเวลาว่าถึงนาทีที่

0 แล้วก็ทำการส่งไฟล์ขึ้นเว็บไซต์

string HostFile = ea\_name + str + ".csv";

success=FtpPutFileW(hIntObjConn,

LocalFile, HostFile,

FTP\_TRANSFER\_TYPE\_BINARY, NULL);

} //end if

FileDatetime(); // ฟังค์ชั่นอ่านค่าวันและเวลา

} //end if

else if(Days == Arr\_Date[2] && hour != "0"){

// เงื่อนไขตรวจสอบว่ามีไฟล์ที่เขียนขึ้นมาแล้ว

printf("Day %s == Arr\_Date %s && hour %s",

Days, Arr\_Date[2], hour);

if(Minute() == 30 || Minute() == 0){

// ตรวจสอบว่าทุก 30 นาทีให้ทำการส่งไฟล์ขึ้นเว็บไซต์

printf("Minute = %d", Minute()); // แสดงเวลา

handle = FileOpen(

ea\_name + str + ".csv", FILE\_CSV |

FILE\_READ | FILE\_WRITE, ',');

if(FileSeek(handle, 0, SEEK\_END) == true){

// ตรวจสอบบรรทัดสุดท้ายของการเขียนไฟล์

printf("FileSeek");

FileWrite(handle, TimeToStr(

TimeLocal(), TIME\_SECONDS),

DoubleToStr(Money, 2),

DoubleToStr(AccountBalance(), 2),

DoubleToStr(total, 0),

DoubleToStr(Buy, 0),

DoubleToStr(Sell, 0),

DoubleToStr(NumWin, 0),

DoubleToStr(Buffer, 2),

DoubleToStr(Arr\_Loss[1], 2),

DoubleToStr(Arr\_Loss[2], 2),

DoubleToStr(Arr\_Loss[3], 2),

DoubleToStr(Arr\_Loss[4], 2));

FileClose(handle);

string LocalFile = TerminalInfoString(

TERMINAL\_DATA\_PATH) +

"\\MQL4\\Files\\" +

ea\_name + str + ".csv";

if(Minute() == 0 || Minute() == 30){

string HostFile = ea\_name +

str +".csv";

success = FtpPutFileW(

hIntObjConn, LocalFile,

HostFile,

FTP\_TRANSFER\_TYPE\_BINARY,

NULL);

} //end if

} //end if

} //end else if

FileDatetime();

} //end if

} //end if

} // end if

InternetCloseHandle(hIntObj);

if(success == false){ // ตรวจสอบว่าส่งไฟล์ขึ้นเว็บไซต์สำเร็จ

printf("Downloading/Uploading error. %d", success);

}else{ // ตรวจสอบว่าส่งไฟล์ขึ้นเว็บไซต์ไม่สำเร็จ

printf("Downloading/Uploading sucessfull!!");

Days = Arr\_Date[2];

} //end else

}else{ //ตรวจสอบในกรณีที่ผู้ใช้ทำการทดสอบย้อนหลัง

handle = FileOpen(ea\_name + str + ".csv", FILE\_CSV |

FILE\_READ | FILE\_WRITE, ',');

Check\_Time();

FileDatetime();

if(Days != Arr\_Date[2]){

FileWrite(handle, "Time", "Money", "Balance",

"Position", "Buy", "Sell", "StatWin", "Buffer",

"Layer1", "Layer2", "Layer3", "Layer4", "\n"

+ TimeToStr(TimeLocal(), TIME\_SECONDS),

DoubleToStr(Money, 2),

DoubleToStr(AccountBalance(), 2),

DoubleToStr(total, 0),

DoubleToStr(Buy, 0),

DoubleToStr(Sell, 0),

DoubleToStr(NumWin, 0),

DoubleToStr(Buffer, 2),

DoubleToStr(Arr\_Loss[1], 2),

DoubleToStr(Arr\_Loss[2], 2),

DoubleToStr(Arr\_Loss[3], 2),

DoubleToStr(Arr\_Loss[4], 2));

FileClose(handle);

string LocalFile=TerminalInfoString(

TERMINAL\_DATA\_PATH) + "\\tester\\files\\" +

ea\_name + str +".csv";

FileDatetime();

Days = Arr\_Date[2];

}else if(Days == Arr\_Date[2] && hour != "0"){

if(Minute() == 30 || Minute() == 0){

handle=FileOpen(ea\_name + str + ".csv",

FILE\_CSV | FILE\_READ | FILE\_WRITE, ',');

if(FileSeek(handle, 0, SEEK\_END) == true){

//Check Line

FileWrite(handle, TimeToStr(

TimeLocal(), TIME\_SECONDS),

DoubleToStr(Money, 2),

DoubleToStr(AccountBalance(), 2),

DoubleToStr(total, 0),

DoubleToStr(Buy, 0),

DoubleToStr(Sell, 0),

DoubleToStr(NumWin, 0),

DoubleToStr(Buffer,2),

DoubleToStr(Arr\_Loss[1], 2),

DoubleToStr(Arr\_Loss[2], 2),

DoubleToStr(Arr\_Loss[3], 2),

DoubleToStr(Arr\_Loss[4], 2));

FileClose(handle);

string LocalFile = TerminalInfoString(

TERMINAL\_DATA\_PATH) +

"\\tester\\files\\" +

ea\_name + str + ".csv";

} //end if

} //end if

FileDatetime();

} //end else if

} //end if

} //end function

//================================================================

void FileDatetime(){ //ฟังค์ชั่น (Function) รับค่าวันที่

Arr\_Date[0] = StringSubstr(TimeToStr(TimeLocal(),

TIME\_DATE), 0, 4); // year

Arr\_Date[1] = StringSubstr(TimeToStr(TimeLocal(),

TIME\_DATE), 5, 2); // month

Arr\_Date[2] = StringSubstr(TimeToStr(TimeLocal(),

TIME\_DATE), 8, 2); // day

str = Arr\_Date[0] + Arr\_Date[1] + Arr\_Date[2];

}

//================================================================

void Check\_Time(){ //ฟังค์ชั่น (Function) ตรวจสอบเวลา

hour = StrToDouble(StringSubstr(TimeToStr(TimeLocal(),

TIME\_SECONDS), 0, 2));

} //end function

//================================================================

void Open\_Pos(){ // ฟังค์ชั่น (Function) ส่งคำสั่งการซื้อขาย

printf("Open\_Pos");

if(Open\_Buy() == 1 && Option() == 1){

// ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้เปิดคำสั่งการซื้อและกำหนดจุดทำกำไรหรือไม่

if(OrderSend(Symbol(), OP\_BUY, Lots, Ask, 3,

Ask-SL\*Point, Ask+TP\*Point, comment, Magic, 0,

clrGreen)){

Status = "Open\_Buy";

Layer = Layer + 1;

Buy = Buy + 1;

total = total + 1;

}else

printf("%d",GetLastError());

} //end if

else if(Open\_Buy() == 1 && Option() == 0){

//ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้เปิดคำสั่งการซื้อเพียงอย่างเดียวหรือไม่

if( OrderSend(Symbol(), OP\_BUY, Lots, Ask, 3,

Ask-SL\*Point, 0, comment, Magic, 0, clrGreen)){

Status = "Open\_Buy NO TP";

Layer = Layer + 1;

Buy = Buy + 1;

total = total + 1;

}else

printf("%d", GetLastError());

}else if(Open\_Sell() == 1 && Option() == 1){

//ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้เปิดคำสั่งการขายและกำหนดจุดทำกำไรหรือไม่

if(OrderSend(Symbol(), OP\_SELL, Lots, Bid, 3,

Bid+SL\*Point, Bid-TP\*Point, comment, Magic, 0,

clrRed)){

Status = "Open\_Sell";

Layer = Layer + 1;

Sell = Sell + 1;

total = total + 1;

}else

printf("%d", GetLastError());

}else if(Open\_Sell() == 1 && Option() == 0){

//ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้เปิดคำสั่งการขายเพียงอย่างเดียวหรือไม่

if( OrderSend(Symbol(), OP\_SELL, Lots, Bid, 3,

Ask+SL\*Point, 0, comment, Magic, 0, clrRed)){

Status = "Open\_Sell NO TP";

Layer = Layer + 1;

Sell = Sell + 1;

total = total + 1;

}else

printf("%d", GetLastError());

} //end else if

} //end function

//================================================================

void Close\_Pos(){ // ฟังค์ชั่น (Function) ปิดคำสั่งการซื้อขาย

if(Close\_Buy() == 1 && Option() == 0){ // ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้ปิดคำสั่งการซื้อด้วยเครื่องมือบ่งชี้ (Indicator) เพียงอย่างเดียวหรือไม่

Status = "Close\_Buy";

printf("Close\_Buy");

for(int i = 0; i < OrdersTotal(); i++){ //วนรอบเพื่อหาคำสั่งการขายและปิดคำสั่ง

if(OrderSelect(i,SELECT\_BY\_POS,MODE\_TRADES) == 1){

if(OrderType() == OP\_BUY &&

Bid > OrderOpenPrice()){

if (OrderClose(OrderTicket(), Lots, Bid, 3,

clrAliceBlue)){ // ปิดคำสั่งการซื้อขาย

Stat\_Close\_Pos = 1;

NumWin = NumWin + 1;

}else

printf("%d",GetLastError());

} //end if

} //end if

} //end if

}else if(Close\_Buy() == 1 && Option() == 1){ // ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้ปิดคำสั่งการซื้อด้วยเครื่องมือบ่งชี้ (Indicator) และกำหนดจุดการปิดหรือไม่

Status = "Close\_Buy & Option";

printf("Close\_Buy & Option");

for(int i = 0; i < OrdersTotal(); i++){ //วนรอบเพื่อหาคำสั่งการขายและปิดคำสั่ง

if(OrderSelect(i, SELECT\_BY\_POS, MODE\_TRADES) == 1){

if(OrderType() == OP\_BUY &&

Bid > OrderOpenPrice()){ // ตรวจสอบว่ารายได้ไม่มีการติดลบ

if(OrderClose(OrderTicket(), Lots, Bid, 3,

clrAliceBlue)){

Stat\_Close\_Pos = 1;

NumWin = NumWin + 1;

}else

printf("%d", GetLastError());

} //end if

} //end if

} //end else if

}else if(Close\_Sell() == 1 && Option() == 0){ // ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้ปิดคำสั่งการขายด้วยเครื่องมือบ่งชี้ (Indicator) เพียงอย่างเดียวหรือไม่

Status = "Close\_Sell";

printf("Close\_Sell");

for(int i = 0; i < OrdersTotal(); i++){ //วนรอบรับค่าเพื่อหาคำสั่งการซื้อขาย

if(OrderSelect(i, SELECT\_BY\_POS, MODE\_TRADES) == 1){

if(OrderType() == OP\_SELL &&

Ask < OrderOpenPrice()){ // ตรวจสอบว่ารายได้ไม่มีการติดลบ

if(OrderClose(OrderTicket(), Lots, Ask, 3,

clrAliceBlue)){

Stat\_Close\_Pos = 1;

NumWin = NumWin + 1;

//DownLoadUploadFile();

}else

printf("%d", GetLastError());

} //end if

} //end if

} // end for

}else if(Close\_Sell() == 1 && Option() == 1){ // ตรวจสอบว่าผู้ใช้ได้ทำการป้อนตรรกะให้ปิดคำสั่งการขายด้วยเครื่องมือบ่งชี้ (Indicator) และกำหนดจุดการปิดหรือไม่

Status = "Close\_Sell & Option";

printf("Close\_Sell & Option");

for(int i = 0; i < OrdersTotal(); i++){ // วนรอบหาคำสั่งการซื้อขาย

if(OrderSelect(i,SELECT\_BY\_POS,MODE\_TRADES) == 1){

if( OrderType() == OP\_SELL &&

Ask < OrderOpenPrice()){

if(OrderClose(OrderTicket(), Lots, Ask, 3,

clrAliceBlue)){

Stat\_Close\_Pos = 1;

NumWin = NumWin + 1;

}else

printf("%d",GetLastError());

} //end if

} //end if

} //end for

} //end else if

} //end function

//================================================================

void Cal\_Layer(){ //ฟังค์ชั่น (Function) คำนวณการบริหารเงินแบบชั้น

if(OrdersHistoryTotal() > 0){ //ตรวจสอบว่ามีการส่งคำสั่งการซื้อขายไปแล้วหรือยัง

Status = "";

int check = OrdersHistoryTotal() - 1;

if(OrderSelect(check, SELECT\_BY\_POS, MODE\_HISTORY)){

if((AccountBalance() >= LastMoney &&

OrderMagicNumber() == Magic) ||

(Stat\_Close\_Pos == 1 &&

OrderMagicNumber() == Magic)){

// ตรวจสอบว่าเงินในบัญชีปัจจุบันมากกว่าเงินที่มีอยู่ในบัญชีก่อนหน้านี้

takeprofit = OrderProfit() + OrderSwap() +

OrderCommission();

if( Layer == 1 && Buffer >= 0 ){ // การส่งคำสั่งการซื้อขายอยู่ในการบริหารเงินชั้นที่ 1 และยังไม่มีเงินติดลบ

printf("Layer = 1 & Buffer >= 0 [TP]");

printf("takeprofit: %f",takeprofit);

Buffer = Buffer + takeprofit;

Layer = Layer - 1;

CheckOrders = OrdersHistoryTotal();

Stat\_Close\_Pos = 0;

NumWin = NumWin + 1;

}else if(Layer > 1 &&

(takeprofit + Buffer) >=

MathAbs(Arr\_Loss[Layer-1])){ // การส่งคำสั่งการซื้อขายอยู่ในการบริหารเงินมากกว่าชั้นที่ 1และกำไรมากกว่าเงินที่ติดลบอยู่ในชั้นข้างบน

if(Layer > 2 &&

(takeprofit + Buffer) >=

MathAbs(Arr\_Loss[Layer-1] +

Arr\_Loss[Layer-2]) &&

MathAbs(Arr\_Loss[1] != 0){ // การส่งคำสั่งการซื้อขายอยู่ในการบริหารเงินมากกว่าชั้นที่ 1 และกำไรมากกว่าเงินที่ติดลบอยู่ในชั้นข้างบน

printf("Layer > 1 & Buffer <

Before TWO Layer [TP]");

printf("takeprofit: %f",

takeprofit); // แสดงกำไรของคำสั่งการซื้อขาย

Buffer = ( takeprofit + Buffer ) +

(Arr\_Loss[Layer - 1] +

Arr\_Loss[Layer - 2]);

// นำกำไรมาหกลบกับจำนวนเงินที่ติดลบอยู่

Arr\_Loss[Layer-1] = 0;

Arr\_Loss[Layer-2] = 0;

Layer = Layer - 3;

CheckOrders = OrdersHistoryTotal();

Stat\_Close\_Pos = 0;

NumWin = NumWin + 1;

printf("Layer: %d",Layer);

}else{

printf("Layer > 1 & Buffer >

Before Layer [TP]");

printf("takeprofit: %f", takeprofit);

Buffer = ( takeprofit + Buffer ) +

Arr\_Loss[Layer - 1];

Arr\_Loss[Layer-1] = 0;

Layer = Layer - 2;

CheckOrders = OrdersHistoryTotal();

Stat\_Close\_Pos = 0;

NumWin = NumWin + 1;

printf("Layer: %d",Layer);

} //end else

}else if(Layer > 1 && ( takeprofit + Buffer ) <

MathAbs(Arr\_Loss[Layer-1])){ // ส่งคำสั่งการซื้อขายอยู่ในการบริหารเงินมากกว่าชั้นที่ 1 และ กำไรรวมกับเงินสำรองที่มีอยู่มีค่าน้อยกว่าเงินที่ติดลบอยู่ในชั้นข้างบน

printf("Layer > 1 & Buffer < Before Layer [TP]");

printf("takeprofit: %f",takeprofit);

Buffer = Buffer + takeprofit;

Layer = Layer - 1;

CheckOrders = OrdersHistoryTotal();

Stat\_Close\_Pos = 0;

NumWin = NumWin + 1;

} //end else if

}else if(AccountBalance() < LastMoney &&

OrderMagicNumber() == Magic){

// ตรวจสอบว่าเงินในบัญชีปัจจุบันมีค่าน้อยกว่าเงินที่มีก่อนหน้านี้

stoploss = OrderProfit() + OrderSwap() +

OrderCommission(); // หาค่าจำนวนเงินสุทธิที่โดนตัดขาดทุน

if(Layer == 1 && Buffer >= MathAbs(stoploss)){ // การส่งคำสั่งการซื้อขายอยู่ในการบริหารเงินชั้นที่ 1 และเงินสำรองมากกว่าเงินที่โดนตัดขาดทุน

printf("Layer = 1 & Buffer > SL [SL]");

printf("stoploss: %f",takeprofit);

// แสดงจำนวนเงินที่โดนตัดขาดทุน

Buffer = Buffer + stoploss;

Layer = Layer - 1;

CheckOrders = OrdersHistoryTotal();

}else if(Layer == 1 &&

Buffer < MathAbs(stoploss)){ // การส่งคำสั่งการซื้อขายอยู่ในการบริหารเงินชั้นที่ 1 และเงินสำรองน้อยกว่าเงินที่โดนตัดขาดทุน

printf("Layer = 1 & Buffer < SL [SL]");

printf("stoploss: %f",takeprofit);

// แสดงจำนวนเงินที่โดนตัดขาดทุน

Arr\_Loss[Layer] = stoploss;

CheckOrders = OrdersHistoryTotal();

}else if(Layer > 1 &&

Buffer < MathAbs(stoploss)){ // การส่งคำสั่งการซื้อขายอยู่ในการบริหารเงินมากกว่าชั้นที่ 1 และเงินสำรองน้อยกว่าเงินที่โดนตัดขาดทุน

printf("Layer > 1 & Buffer < SL [SL]");

printf("stoploss: %f",takeprofit);

// แสดงจำนวนเงินที่โดนตัดขาดทุน

Arr\_Loss[Layer] = stoploss;

CheckOrders = OrdersHistoryTotal();

} //end else if

} //end else if

} //end else

} //end if

} //end function

//================================================================

bool IsBarClosed(int timeframe, bool reset){ // ฟังค์ชั่นส่งคำสั่งหลังจากจบแท่งเทียน

static datetime lastbartime;

if(timeframe == -1){

if(reset)

lastbartime = 0;

else

lastbartime = iTime(NULL, timeframe, 0);

return(true);

} //end if

if(iTime(NULL, timeframe, 0) == lastbartime) // wait for new bar

return(false);

if(reset)

lastbartime = iTime(NULL, timeframe, 0);

return(true);

} //end function

//================================================================

int OnInit(){

Money = AccountBalance();

IsBarClosed(-1, false);

FileDatetime();

CheckOrders = OrdersHistoryTotal();

for(int i = 0; i < OrdersTotal(); i++){

if(OrderSelect(i, SELECT\_BY\_POS, MODE\_TRADES)){

if(OrderMagicNumber() == Magic){

printf("Power Off");

LastPosition = OrdersTotal();

comment = OrderComment();

LastMoney = AccountBalance();

printf("Comment: %s", comment);

} //end if

} //end if

} //end for

return(INIT\_SUCCEEDED);

} //end function

//================================================================

int start(){

if(LastPosition > 0){ // ตรวจสอบว่ามีการส่งคำสั่งการซื้อขายหรือไม่

CheckOrders = OrdersHistoryTotal();

Layer = StrToDouble(comment);

printf("LastPosition: %d", Layer);

printf("Layer: %d", Layer);

} //end if

comment = DoubleToStr(Layer+1, 0);

if(!IsBarClosed(0, true))

return(0);

if(Layer < 5){ // ตรวจสอบว่าชั้นการบริหารเงินนั้นเกินชั้นที่ 4 หรือไม่

if(AccountEquity() != 0 && Layer < 5){

if(OrdersTotal() == 0 &&

CheckOrders == OrdersHistoryTotal()){

LastMoney = AccountBalance();

Open\_Pos(); // ฟังค์ชั่น (Function) เปิดคำสั่งการซื้อขาย

DownLoadUploadFile();

// ฟังค์ชั่น (Function) ทำการส่งไฟล์ขึ้นเว็บไซต์

}else if(OrdersTotal() == 1 &&

CheckOrders == OrdersHistoryTotal()){

Close\_Pos(); // ฟังค์ชั่น (Function) ปิดคำสั่งการซื้อขาย

DownLoadUploadFile();

// ฟังค์ชั่น (Function) ทำการส่งไฟล์ขึ้นเว็บไซต์

}else if(OrdersTotal() == 0 &&

CheckOrders != OrdersHistoryTotal()){

Cal\_Layer(); // ฟังค์ชั่นคำนวณการบริหารเงินแบบชั้น

DownLoadUploadFile(); // ฟังค์ชั่นทำการส่งไฟล์ขึ้นเว็บไซต์

} //end else if

} //end if

LastPosition = 0;

}else{

for(int i = 0; i < OrdersTotal(); i++){ // วนรอบปิดคำสั่งการซื้อขายทั้งหมด

if(OrderSelect(i, SELECT\_BY\_POS, MODE\_TRADES)){

if(OrderType() == OP\_BUY){ // ถ้าเป็นคำสั่งการซื้อ

OrderClose(OrderTicket(), Lots, Ask,

3, clrBlue);

OrderClose(OrderTicket(), Lots, Bid,

3, clrBlue);

}else if(OrderType() == OP\_SELL){ // ถ้าเป็นคำสั่งการขาย

OrderClose(OrderTicket(), Lots, Ask,

3, clrBlue);

OrderClose(OrderTicket(), Lots, Bid,

3, clrBlue);

} //else if

} //end if

} //end for

printf("Error");

} //end else

DownLoadUploadFile(); // ฟังค์ชั่น (Function) ทำการส่งไฟล์ขึ้นเว็บไซต์

Comment(" \n"

+" Balance: " + DoubleToStr(AccountBalance(), 2) // แสดงจำนวนเงินสุทธิ

+" \n"

+" Money: " + DoubleToStr(Money, 2) // แสดงจำนวนเงินที่เติมครั้งแรก

+" \n"

+" Equity : " + DoubleToStr(AccountEquity(), 2) // แสดงการเคลื่อนไหวของเงิน

+" \n"

+" Layer : " + IntegerToString(Layer, 0) // แสดงจำนวนชั้นการบริหารเงิน

+" \n"

+" Status : " + Status // แสดงสถานะคำสั่งการซื้อขาย

+" \n"

+" OrdersTotoal : " + DoubleToStr(OrdersTotal(), 0) // แสดงจำนวนคำสั่งการซื้อขายทั้งหมด

+" \n"

+" Total : " + DoubleToStr(total, 0) // แสดงจำนวนคำสั่งการซื้อขายทั้งหมด

+" \n"

+" Buy : " + DoubleToStr(Buy, 0) // แสดงจำนวนชนิดของคำสั่งการซื้อ

+" Sell : " + DoubleToStr(Sell, 0) // แสดงจำนวนชนิดของคำสั่งการขาย

+" \n"

+" Win : " + DoubleToStr(NumWin, 0) // แสดงจำนวนคำสั่งการซื้อขายที่ทำกำไรได้

+" \n"

+" Buffer : " + DoubleToStr(Buffer, 2) // แสดงจำนวนเงินสำรอง

+" \n"

+" Layer1 : " + DoubleToStr(Arr\_Loss[1], 2) // แสดงจำนวนเงินที่ขาดทุนในชั้นที่ 1

+" \n"

+" Layer2 : " + DoubleToStr(Arr\_Loss[2], 2) // แสดงจำนวนเงินที่ขาดทุนในชั้นที่ 2

+" \n"

+" Layer3 : " + DoubleToStr(Arr\_Loss[3], 2) // แสดงจำนวนเงินที่ขาดทุนในชั้นที่ 3

+" \n"

+" Layer4 : " + DoubleToStr(Arr\_Loss[4], 2) // แสดงจำนวนเงินที่ขาดทุนในชั้นที่ 4

+" \n");

return (0);

} //end start