
	Politechnika Bydgoska im. J. J. Śniadeckich Wydział Telekomunikacji, Informatyki i Elektrotechniki		
Przedmiot	Analiza Regresji i Szeregów Czasowych		
Prowadzący	dr inż. Damian Ledziński		
Temat	JForex podstawy i strategię		
Student	Cezary Tytko		
Nr lab.	1 i 2	Data wykonania	9.05.2024
Ocena		Data oddania spr.	

JForex

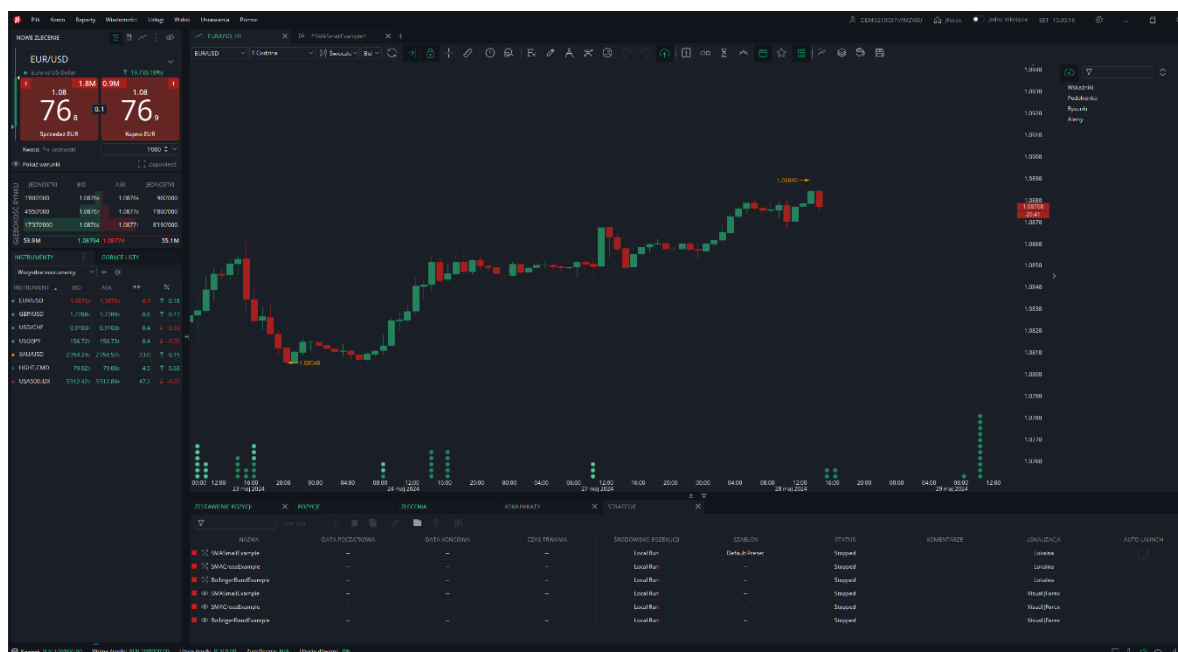
JForex to zaawansowana platforma handlowa stworzona przez firmę Dukascopy Bank SA, przeznaczona do handlu na rynku Forex oraz kontraktami różnic kursowych (CFD). JForex jest szczególnie popularna wśród profesjonalnych traderów oraz tych, którzy stosują automatyczne strategię handlowe.

Forex

Forex, znany również jako rynek walutowy, jest globalnym rynkiem zdecentralizowanym przeznaczonym do handlu walutami. Jest to największy i najbardziej płynny rynek finansowy na świecie, z dziennym obrotem przekraczającym 6 bilionów dolarów amerykańskich.

Rynek Forex umożliwia uczestnikom – w tym bankom, instytucjom finansowym, korporacjom, rządowi oraz inwestorom indywidualnym – wymianę jednej waluty na inną. Handel walutami odbywa się w parach, gdzie jedna waluta jest kupowana, a druga sprzedawana. Najbardziej popularne pary walutowe obejmują EUR/USD, GBP/USD, USD/JPY i wiele innych.

Program posiada mnóstwo narzędzi, najciekawsze z nich i najprzydatniejsze wymienięm poniżej



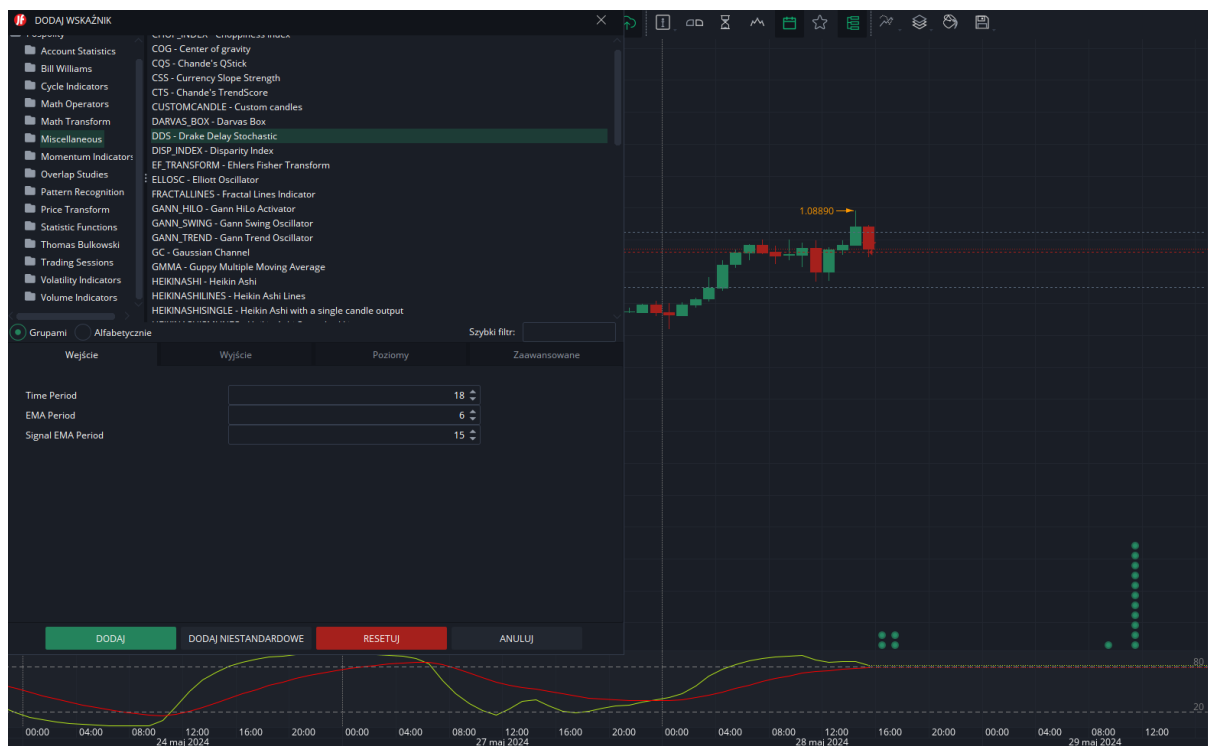
Ogólnie platforma umożliwia tworzenie i uruchamianie zautomatyzowanych strategii handlowych za pomocą języka programowania Java. Traderzy mogą tworzyć własne algorytmy, testować je na danych historycznych oraz uruchamiać je w czasie rzeczywistym.

ZESTAWIENIE POZYCJI (1)		POZYCJE (1)		ZŁYCENIA		KOMUNIKATY		STRATEGIE		
NR POZYCJI	INSTRUMENT	KIERUNEK	KWOTA	CENA	OBECNA CENA	STOP LOSS	TAKE PROFIT	ZYS W PIPSACH	ZYS	KOMENTARZ
236654755	EUR/USD	SHORT	EUR 1000	1.08761	1.08751			6.7	PLN 0.27	

Widok aktualnie zawartych transakcji



Transakcja pokazana na wykresie z ustalonymi progami take profit i stop loss



JForex oferuje szeroki zakres wskaźników technicznych, narzędzi do analizy graficznej oraz zaawansowane funkcje tworzenia wykresów, które pomagają w podejmowaniu świadomych decyzji inwestycyjnych.

EUR/USD, H1

*SMASmallExample1

+

Obfuscacja

?

```

1 package examples;
2
3 import java.util.*;
4 import java.text.*;
5
6 import com.dukascopy.api.*;
7
8 public class SMASmallExample1 implements IStrategy {
9     private IEngine engine;
10    private IConsole console;
11    private IHistory history;
12    private IContext context;
13    private IIndicators indicators;
14    private IUserInterface userInterface;
15
16    @Configurable("selectedInstrument:")
17    public Instrument selectedInstrument = Instrument.EURUSD;
18    @Configurable("tradeAmount:")
19    public double tradeAmount = 0.0010;
20    @Configurable("slippage:")
21    public int slippage = 5;
22    @Configurable("stopLossPips:")
23    public int stopLossPips = 25;
24    @Configurable("takeProfitPips:")
25    public int takeProfitPips = 50;
26
27    private static final DateFormat DATE_FORMAT = new SimpleDateFormat("yyyyMMdd_HH:mm:ss");
28
29    public void onStart(IContext context) throws JFException {
30        this.engine = context.getEngine();
31        this.console = context.getConsole();
32        this.history = context.getHistory();
33        this.context = context;
34        this.indicators = context.getIndicators();
35        this.userInterface = context.getUserInterface();
36
37        context.setSubscribedInstruments(Collections.singleton(selectedInstrument), true);
38    }
39
40    public void onAccount(IAccount account) throws JFException {
41    }
42
43    public void onMessage(IMessage message) throws JFException {
44    }
45
46    public void onStop() throws JFException {
47    }
48
49    public void onTick(Instrument instrument, ITick tick) throws JFException {
50        if (!instrument.equals(selectedInstrument)) {
51            return;
52        }
53
54        if (!engine.getOrders(selectedInstrument).isEmpty()) {

```

ZESTAWIENIE POZYCJI (1)

POZYCJE (1)

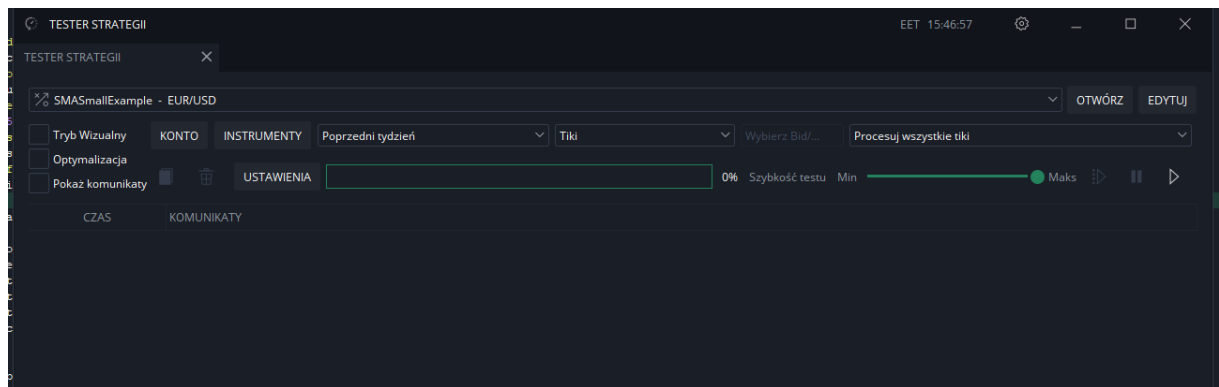
ZLECENIA (2)

KOMUNIKATY

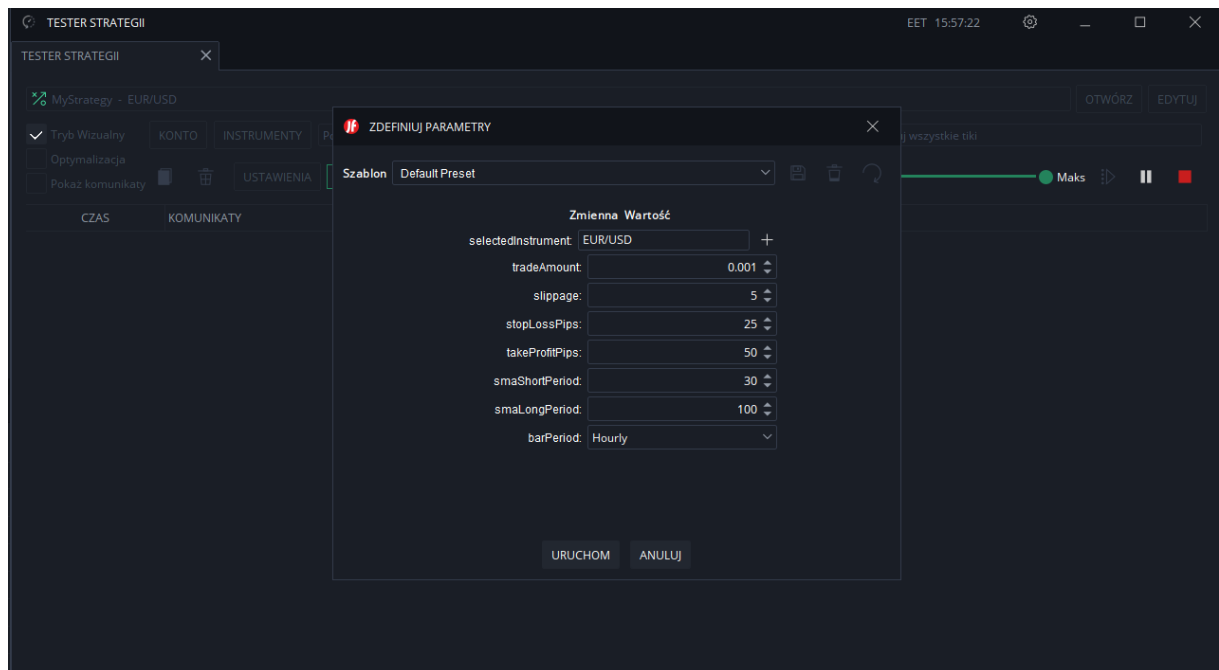
Local Run

NAZWA	DATA POCZĄTKOWA	DATA KOŃCOWA	CZAS TRWANIA
SMASmallExample	--	--	--
SMACrossExample	--	--	--
BollingerBandExample	--	--	--
SMASmallExample	--	--	--
SMACrossExample	--	--	--
BollingerBandExample	--	--	--

Platforma umożliwia tworzenie i uruchamianie zautomatyzowanych strategii handlowych za pomocą języka programowania Java. Traderzy mogą tworzyć własne algorytmy, testować je na danych historycznych oraz uruchamiać je w czasie rzeczywistym.



Możliwość testowania strategii na danych historycznych



Dynamiczne ustawianie parametrów, dla odpowiednio oznaczonych pól w definicji strategii.

Własna strategia:

```
1. package jforex;
2.
3. import java.util.*;
4. import java.text.*;
5.
6. import com.dukascopy.api.*;
7.
8. public class MyStrategy implements IStrategy {
9.     private IEngine engine;
10.    private IConsole console;
11.    private IHistory history;
12.    private IContext context;
13.    private IIndicators indicators;
14.    private IUserInterface userInterface;
15.
16.    @Configurable("selectedInstrument:")
17.    public Instrument selectedInstrument = Instrument.EURUSD;
18.    @Configurable("tradeAmount:")
19.    public double tradeAmount = 0.0010;
20.    @Configurable("slippage:")
21.    public int slippage = 5;
22.    @Configurable("stopLossPips:")
23.    public int stopLossPips = 25;
24.    @Configurable("shortSMAPeriod:")
25.    public int shortSMAPeriod = 30;
26.    @Configurable("longSMAPeriod:")
27.    public int longSMAPeriod = 100;
28.    @Configurable("barPeriod:")
29.    public Period barPeriod = Period.ONE_HOUR;
30.
31.    private static final DateFormat DATE_FORMAT = new SimpleDateFormat("yyyyMMdd_HH:mm:ss");
32.    private Map<IOOrder, Long> orderOpenTimes = new HashMap<>();
33.
34.    public void onStart(IContext context) throws JFException {
35.        this.engine = context.getEngine();
36.        this.console = context.getConsole();
37.        this.history = context.getHistory();
38.        this.context = context;
39.        this.indicators = context.getIndicators();
40.        this.userInterface = context.getUserInterface();
41.
42.        context.setSubscribedInstruments(Collections.singleton(selectedInstrument), true);
43.    }
44.
45.    public void onAccount(IAccount account) throws JFException {
46.    }
47.
48.    public void onMessage(IMessage message) throws JFException {
49.    }
50.
51.    public void onStop() throws JFException {
52.    }
53.
54.    public void onTick(Instrument instrument, ITick tick) throws JFException {
55.    }
56.
57.    public void onBar(Instrument instrument, Period period, IBar askBar, IBar bidBar) throws
JFException {
58.        if (!instrument.equals(selectedInstrument) || !period.equals(barPeriod)) {
59.            return;
60.        }
61.
62.        long time = history.getBar(selectedInstrument, barPeriod, OfferSide.BID, 0).get-
Time();
63.        Object[] indicatorResult = indicators.calculateIndicator(selectedInstrument, Pe-
riod.TEN_MINS, new OfferSide[] {OfferSide.BID},
```

```

64.         "SMA", new IIndicators.AppliedPrice[] {IIndicators.AppliedPrice.CLOSE}, new
Object[] {30}, 0);
65.         double smaSmallCurrent = (Double) indicatorResult[0];
66.
67.         indicatorResult = indicators.calculateIndicator(selectedInstrument, Period.TEN_MINS,
new OfferSide[] {OfferSide.BID},
68.         "SMA", new IIndicators.AppliedPrice[] {IIndicators.AppliedPrice.CLOSE}, new
Object[] {30}, 1);
69.         double smaSmallPrev = (Double) indicatorResult[0];
70.
71.         indicatorResult = indicators.calculateIndicator(selectedInstrument, Period.ONE_HOUR,
new OfferSide[] {OfferSide.BID},
72.         "SMA", new IIndicators.AppliedPrice[] {IIndicators.AppliedPrice.CLOSE}, new
Object[] {30}, 0);
73.         double smaBigCurrent = (Double) indicatorResult[0];
74.
75.         indicatorResult = indicators.calculateIndicator(selectedInstrument, Period.ONE_HOUR,
new OfferSide[] {OfferSide.BID},
76.         "SMA", new IIndicators.AppliedPrice[] {IIndicators.AppliedPrice.CLOSE}, new
Object[] {30}, 1);
77.         double smaBigPrev = (Double) indicatorResult[0];
78.
79.         IOrder order = getOpenOrder(selectedInstrument);
80.
81.         if (order == null) {
82.             if (smaSmallCurrent > smaBigCurrent && smaBigCurrent < smaBigPrev) {
83.                 double stopLoss = bidBar.getClose() - selectedInstrument.getPipValue() *
stopLossPips;
84.                 IOrder newOrder = engine.submitOrder(getLabel(time), selectedInstrument,
IEngine.OrderCommand.BUY,
85.                 tradeAmount, 0, slippage, stopLoss, 0, 0, "");
86.                 orderOpenTimes.put(newOrder, time);
87.             } else if (smaSmallCurrent < smaBigCurrent && smaBigCurrent > smaBigPrev) {
88.                 double stopLoss = askBar.getClose() + selectedInstrument.getPipValue() *
stopLossPips;
89.                 IOrder newOrder = engine.submitOrder(getLabel(time), selectedInstrument,
IEngine.OrderCommand.SELL,
90.                 tradeAmount, 0, slippage, stopLoss, 0, 0, "");
91.                 orderOpenTimes.put(newOrder, time);
92.             }
93.         } else {
94.             long orderOpenTime = orderOpenTimes.get(order);
95.             if ((time - orderOpenTime) >= barPeriod.getInterval() * 5) {
96.                 if (order.getOrderCommand() == IEngine.OrderCommand.BUY && smaSmallCurrent <
smaBigCurrent) {
97.                     order.close();
98.                     orderOpenTimes.remove(order);
99.                 } else if (order.getOrderCommand() == IEngine.OrderCommand.SELL && sma-
SmallCurrent > smaBigCurrent) {
100.                     order.close();
101.                     orderOpenTimes.remove(order);
102.                 }
103.             }
104.         }
105.     }
106.
107.     private IOrder getOpenOrder(Instrument instrument) throws JFException {
108.         for (IOrder order : engine.getOrders(instrument)) {
109.             if (order.getState() == IOrder.State.FILLED) {
110.                 return order;
111.             }
112.         }
113.         return null;
114.     }
115.
116.     private String getLabel(long time) {
117.         return "SMA" + DATE_FORMAT.format(time) + generateRandom(10000) + generateRan-
dom(10000);
118.     }
119.

```

```

120.     private String generateRandom(int n) {
121.         int randomNumber = (int) (Math.random() * n);
122.         String answer = "" + randomNumber;
123.         if (answer.length() > 3) {
124.             answer = answer.substring(0, 4);
125.         }
126.         return answer;
127.     }
128. }
129.

```

Możemy pisać własne strategie, korzystając z biblioteki „dukascopy.api”, która zawiera implementację pozwalającą otwierać transakcję, jak i narzędzia do analizy stanu rynku, na podstawie których podejmujemy decyzje.

Strategia MyStrategy opiera się na wskaźnikach średnich kroczących (SMA) i działa na bazie danych zebranych w okresach barowych. Strategia otwiera transakcje w oparciu o sygnały generowane przez przecięcia krótkoterminowej i długoterminowej średniej kroczącej, a zamyka je, gdy trend się odwraca i transakcja jest otwarta przynajmniej przez 5 okresów (barów).

Kluczowe elementy strategii:

Parametry konfiguracyjne:

selectedInstrument: Instrument, na którym strategia jest uruchamiana (domyślnie EUR/USD).

tradeAmount: Wielkość transakcji (domyślnie 0.0010).

slippage: Akceptowany poślizg cenowy (domyślnie 5 pipsów).

stopLossPips: Wielkość zlecenia stop loss w pipsach (domyślnie 25).

shortSMAPeriod: Okres krótkoterminowej średniej kroczącej (domyślnie 30).

longSMAPeriod: Okres długoterminowej średniej kroczącej (domyślnie 100).

barPeriod: Okres barów używanych do analizy (domyślnie jedna godzina).

Zasady otwierania zleceń:

Kupno: Jeśli krótkoterminowa SMA przecina długoterminową SMA od dołu do góry, a długoterminowa SMA jest poniżej swojej poprzedniej wartości, otwiera się zlecenie kupna.

Sprzedaż: Jeśli krótkoterminowa SMA przecina długoterminową SMA od góry do dołu, a długoterminowa SMA jest powyżej swojej poprzedniej wartości, otwiera się zlecenie sprzedaży.

Zasady zamykania zleceń:

Kupno: Zlecenie kupna jest zamykane, gdy krótkoterminowa SMA przecina długoterminową SMA od góry do dołu i zlecenie jest otwarte przynajmniej przez 5 okresów.

Sprzedaż: Zlecenie sprzedaży jest zamykane, gdy krótkoterminowa SMA przecina długoterminową SMA od dołu do góry i zlecenie jest otwarte przynajmniej przez 5 okresów.

Pomocnicze metody:

`getOpenOrder()`: Metoda zwracająca otwarte zlecenie dla danego instrumentu.

`getLabel()`: Generuje unikalną etykietę dla zlecenia na podstawie aktualnego czasu.

`generateRandom()`: Generuje losowy numer używany do tworzenia etykiet zleceń.

Podsumowanie

Strategia ta jest prostą, ale NIE efektywną metodą handlu na rynku Forex, która wykorzystuje przecięcia średnich kroczących do identyfikacji sygnałów handlowych i zarządzania pozycjami. Dzięki zastosowaniu dodatkowego warunku, który wymaga, aby transakcja była otwarta przez co najmniej 5 okresów przed zamknięciem, strategia ta stara się unikać zbyt szybkich, impulsywnych zamknięć pozycji, co może zwiększyć jej stabilność i efektywność.

MyStrategy strategy report for EUR/USD instrument(s) from 2022-12-31 22:00:00 to 2023-12-31 21:59:59

Account Currency	USD
Initial Equity	50'000
Final Equity	48'291,58
Turnover	124'327'923
Commission Fees	2669.52

Parameters

selectedInstrument	EUR/USD
tradeAmount	0.1
slippage	5
stopLossPips	10
shortSMAPeriod	20
longSMAPeriod	100
barPeriod	10 Mins

Instrument EUR/USD

First tick time	2022-12-31 22:00:00
First BID	1.07026
First ASK	1.0711
Last tick time	2023-12-29 21:59:59
Last BID	1.10374
Last ASK	1.10393
Positions total	575
Closed positions	574
Orders total	575
Amount bought	57.40
Amount sold	57.50
Turnover	124'327'923
Commission Fees	2669.52

Opened orders:

Label	Amount	Direction	Open price	Profit/Loss at the end	Profit/Loss at the end in pips	Open date	Comment
SMA20231228_18200070754625	0,100	SELL	1.107968	403.8	40.4	2023-12-28 16:20:00	

Closed orders:

Label	Amount	Direction	Open price	Close price	Profit/ Loss	Profit/ Loss in pips	Open date	Close date	Comment
SMA20230101_00100070258221	0,100	SELL	1.07026	1.0711	-84.00	-8.4	2022-12-31 22:10:00	2022-12-31 23:00:00	
SMA20230101_01100025453918	0,100	BUY	1.0711	1.07026	-84.00	-8.4	2022-12-31 23:10:00	2023-01-01 01:00:00	
SMA20230101_03100068036564	0,100	SELL	1.0703157	1.06161	870.57	87.1	2023-01-01 01:10:00	2023-01-04 08:10:00	
SMA20230104_10200072489280	0,100	BUY	1.0618	1.06077	-103.00	-10.3	2023-01-04 08:20:00	2023-01-04 09:06:51	

Strategia została przetestowana na danych z poprzedniego roku dla pary EUR/USD i przyniosła stratę