為替自動売買ツール バックテスト機能 要件リスト

1. プロジェクト目標
- Pythonを用いて、外国為替証拠金取引（FX）の自動売買戦略を検証するためのバックテスト機能を持つフレームワークを構築する（開発の第一段階）。

2. 技術選択
- **バックテストフレームワーク:** Backtrader ライブラリ
- **プログラミング言語:** Python 3.x
- **データ処理:** Pandas ライブラリ
- **データ取得:** Python `MetaTrader5` パッケージ（MT5ターミナル経由）

3. 機能要件

3.1. データハンドリング
- MT5から指定した通貨ペア・時間軸のヒストリカルデータ（日時, OHLCV等）を取得できる。
- 取得データをPandas DataFrameで処理し、CSV等の形式でローカルに保存・管理できる。
- ローカルのデータファイルをBacktraderのデータフィードとして読み込める（`bt.feeds.PandasData`等）。

3.2. 取引戦略定義
- ユーザーが独自の売買シグナル生成ロジック（エントリー/エグジット条件）を定義できる（`bt.Strategy`クラス継承）。
- 各取引の利益確定（TP）/損切り（SL）ルールを設定できる（固定pips, ATR基準等）。

3.3. ロット管理（ポジションサイジング）
- **分解モンテカルロ法**に基づくロット管理機能を実装する。
    - `Strategy`クラス内で現在の「数列」の状態を管理する。
    - 次の取引ロット数は、数列の両端合計（単位ロット数）× 設定可能な「基本単位ロット（`unit_lot`）」で決定する。
    - 取引結果（勝ち/負け）に基づき、`notify_trade`等で数列を更新する（勝ち:両端削除、負け:取引単位ロット数を右端追加）。
    - 数列が1要素になった場合に分解ルールを適用する。
    - 数列が空になった場合にサイクル完了とし、数列をリセットする。
- 基本単位ロット（`unit_lot`）をパラメータとして設定可能にする。

3.4. バックテスト実行エンジン
- 定義されたデータ、戦略、ロット管理ルールに基づき、取引シミュレーションを実行できる（`bt.Cerebro`エンジン）。
- スプレッド、取引手数料を考慮したシミュレーションが可能である。
- （可能であれば）スリッページの影響を簡易的に考慮できる。

3.5. パフォーマンス分析・レポート
- バックテスト完了後、主要なパフォーマンス指標を計算・表示できる:
    - 総損益（Net Profit）
    - プロフィットファクター（Profit Factor）

    - 勝率 (Win Rate)
    - 最大ドローダウン (Max Drawdown)
    - シャープレシオ (Sharpe Ratio)
    - 総トレード数、平均損益など
- 資産曲線、ドローダウン曲線をグラフで可視化できる (`cerebro.plot()`等)。
- （推奨）分解モンテカルロ法のサイクルに関する情報（完了回数、平均期間、サイクル中最大ロット数等）も分析できる。

3.6. リスク管理（検討事項）
- （推奨）最大許容ロット数、最大許容ドローダウンによるセーフティ機能の導入を検討する。

4. 非機能要件（考慮点）
- コードの可読性、保守性。
- 設定変更の容易性。
- エラーハンドリング。