```
//Pin-Bar Sign.mq4

#property indicator_chart_window

#property indicator_buffers 2

#property indicator_color1 Magenta
#property indicator_color2 Aqua

//インジケーターバッファーの宣言
double Arrow_Up[];
double Arrow_Down[];
double Real_Body[];
double Upper_Shadow[];
double Lower_Shadow[];

//変数の宣言
extern int Highest_Period    = 20;
extern int Lowest_Period     = 20;
extern int Magnification     = 3;
extern int Minimum_Length = 30;

double Pips = 0;

//関数の定義
double AdjustPoint(string Currency)
{
  int Symbol_Digits = (int)MarketInfo(Symbol(),MODE_DIGITS);

  if(Symbol_Digits == 2 || Symbol_Digits == 3)
    {
      double Calculated_Point = 0.01;
    }
  else if(Symbol_Digits == 4 || Symbol_Digits == 5)
         {
           Calculated_Point = 0.0001;
         }

  return(Calculated_Point);
}

int init()
{
  IndicatorBuffers(5);

  //インジケーターバッファーのインデックス
  SetIndexBuffer(0,Arrow_Up);
  SetIndexBuffer(1,Arrow_Down);
  SetIndexBuffer(2,Real_Body);
  SetIndexBuffer(3,Upper_Shadow);
  SetIndexBuffer(4,Lower_Shadow);

  //インジケーターのラベル
  SetIndexLabel(0,NULL);
  SetIndexLabel(1,NULL);

  //インジケーターのスタイル
  SetIndexStyle(0,DRAW_ARROW);
  SetIndexArrow(0,233);
  SetIndexStyle(1,DRAW_ARROW);
  SetIndexArrow(1,234);

  //PointとPipsの調整
  Pips = AdjustPoint(Symbol());

  return(0);
}

int start()
{
  int limit = Bars - IndicatorCounted();

  if(Bars < Highest_Period || Bars < Lowest_Period)
    {
      return(0);
    }

  //実体の計算
  for(int i = limit - 1; i >= 0; i--)
     {
       Real_Body[i] = MathAbs(Open[i] - Close[i]);

       if(Real_Body[i] == 0)
         {
           Real_Body[i] = Real_Body[i] + Pips;
         }
     }

  //上ヒゲの計算
  for(i = limit - 1; i >= 0; i--)
     {
       Upper_Shadow[i] = MathMin(High[i] - Open[i],High[i] - Close[i]);
     }

  //下ヒゲの計算
  for(i = limit - 1; i >= 0; i--)
     {
       Lower_Shadow[i] = MathMin(Open[i] - Low[i],Close[i] - Low[i]);
     }

  //矢印の設定
  for(i = limit - 1; i >= 0; i--)
     {
       //上矢印の設定
       if(Real_Body[i] * Magnification <= Lower_Shadow[i] &&
          Minimum_Length * Pips <= Lower_Shadow[i] &&
          Low[i] < Low[iLowest(NULL,0,MODE_LOW,Lowest_Period,i+1)])
        {
          Arrow_Up[i] = Low[i] - Pips;
        }

       //下矢印の設定
       if(Real_Body[i] * Magnification <= Upper_Shadow[i] &&
          Minimum_Length * Pips <= Upper_Shadow[i] &&
          High[i] > High[iHighest(NULL,0,MODE_HIGH,Highest_Period,i+1)])
        {
          Arrow_Down[i] = High[i] + Pips;
        }
     }

  return(0);
}
```

#property命令を記述

インジケーターバッファーを宣言

変数を宣言

関数を定義

基本設定を記述

具体的な処理内容を記述