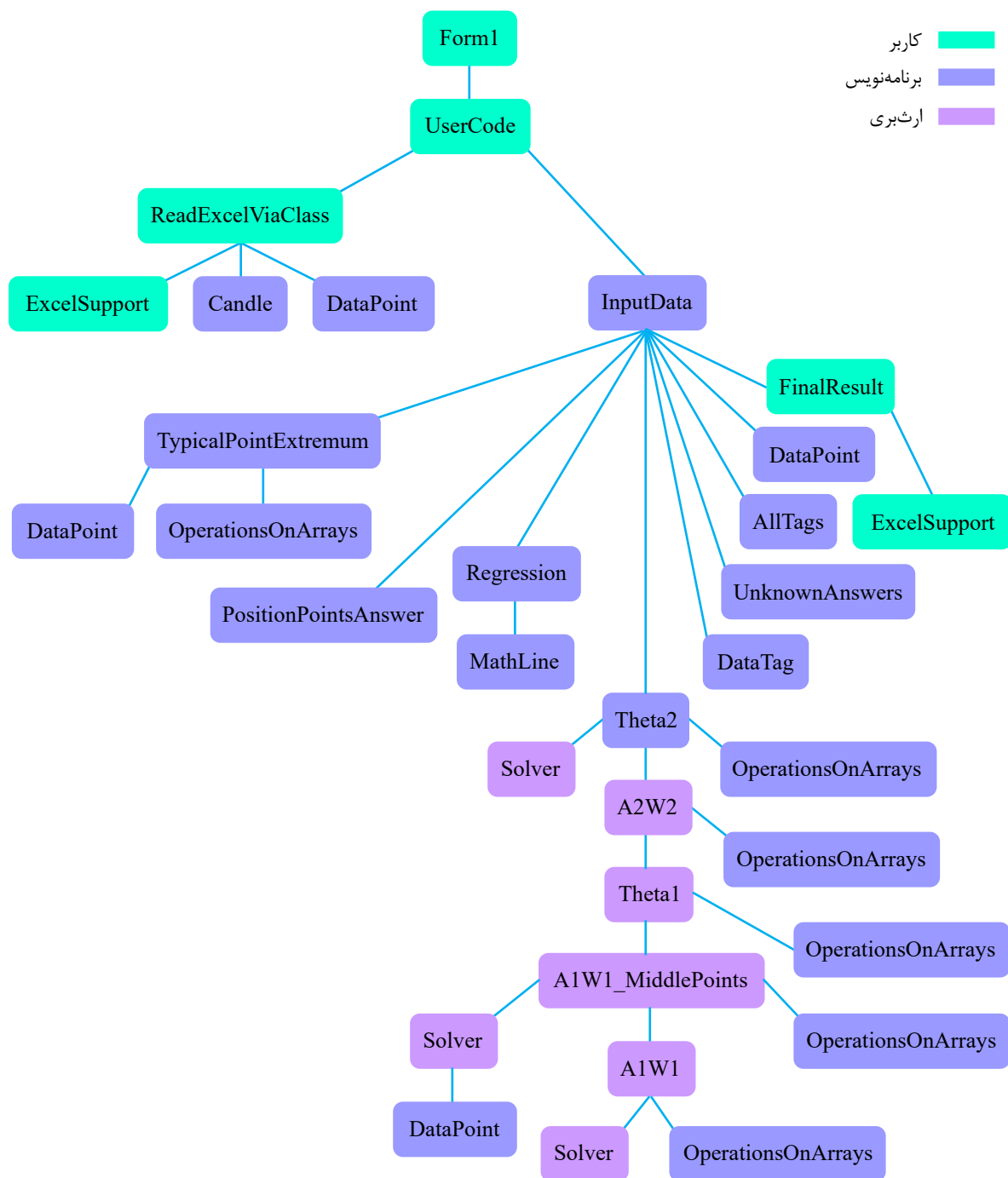


گراف برنامه داتنت به صورت زیر است:



توضیح هر کلاس و متدهای آن:

Form1: در این کلاس اطلاعات مورد نیاز از کاربر گرفته می شود و توسط کاربر این فرم ایجاد می شود. کاربر از متد **Receive_Result** برای بکارگیری کلاس **UserCode** استفاده می کند.

UserCode: کلاسی است که توسط کاربر جهت دادن هر نوع اطلاعاتی به برنامه نویس طراحی می شود. از آنجایی که کاربر از هر نوع فایلی می تواند استفاده کند پس مجبور به تعریف یک سری کلاس های دیگر می شود از جمله کلاس

`ExcelSupport` و `ReadExcelViaClass`. دو کلاس نام برده از طریق متد `ComputePoint` در کلاس `UserCode` بکار گرفته می‌شوند.

`ExcelSupport`: این کلاس برای خواندن و نوشتن سلول به سلول یک اکسل مورد استفاده قرار می‌گیرد. از متد `Get_One_Cell` برای خواندن هر سلول اکسل استفاده می‌شود و از متد `Set_One_Cell` برای نوشتن در هر سلول اکسل استفاده می‌گردد.

`ReadExcelViaClass`: این کلاس برای ذخیره اطلاعات فایل اکسل خوانده شده به کمک کلاس `ExcelSupport` به صورت لیستی از دو کلاس `Candle` و `DataPoint` است. از متد `ComputeAllPoints2` در این کلاس برای ایجاد دو لیست از کلاس `Candle` و `DataPoint` استفاده می‌کنیم.

`Candle`: این کلاس، موجودیت هر کندل را تعریف می‌کند که شامل اطلاعاتی مانند زمان پیوسته، قیمت باز شدن، بالاترین قیمت، پایین‌ترین قیمت، قیمت بسته شدن، حجم در هر کندل، اندیس (زمان گسسته) است.

`DataPoint`: این کلاس، موجودیت هر نقطه را با `constructor`-های متفاوتی، متناسب با نیاز برنامه، تعریف می‌کند. پس از اینکه لیستی از اطلاعات جهت ارائه به برنامه‌نویس آماده گردید، این اطلاعات با فراخوانی کلاس اصلی (از طریق ایجاد شی از آن)، به برنامه‌نویس تحویل داده می‌شود. برنامه اصلی، `InputData` است که برای اجرا این کلاس، ابتدا ورودی به کلاس داده می‌شود و سپس متد مورد نظر (یعنی `Compute`) برای اجرای برنامه از طریق یک شی از این کلاس اجرا می‌شود.

`InputData`: این کلاس، اصلی‌ترین کلاسی است که توسط برنامه‌نویس به کاربر ارائه می‌شود. در این کلاس از کلاس‌ها و متدهای ذیل استفاده می‌شود: `Theta2`, `Regression`, `PositionPointsAnswer`, `TypicalPointExtremum`, `Tags`, `UnknownAnswers`, `DataTag`, `AllTags`, `DataPoint`, `FinalResult`, `CONT_R`, `StabilityCoefficient_DSP` و `Compute`.

کلاس‌های `Theta2`, `Regression`, `TypicalPointExtremum` و `DataTag`، کلاس‌هایی جهت انجام محاسبات هستند و کلاس‌های `PositionPointsAnswer`, `UnknownAnswers`, `AllTags` و `DataPoint`، برای ذخیره محاسبات به صورت یک لیست جهت تحویل دادن به کاربر هستند.

`CONT_R`: برای شمارش تعداد اعضای مخالف صفر یک لیستی از داده‌های `Int` است.

`Tags`: در این متد برای هر 6 نقطه اکسترمم متوالی، اطلاعات زیر کسب می‌شود:

- شماره نقاط اکسترمم دستگاه اول که خط روند از آن دو نقطه رسم شده است.
- تعداد نقاط اکسترممی که در هر دستگاه در بالا یا پایین خط روند رسم شده، قرار گرفته است.
- خط روند رسم شده در هر دستگاه، صعودی است یا نزولی؟
- خط روند رسم شده در هر دستگاه، اگر نزولی است عدد 1 و اگر صعودی است عدد 2 را چاپ کن.
- اگر تمامی نقاط اکسترمم در هر دستگاه در پایین خط روند رسم شده قرار بگیرد `down` خواهد بود و اگر تمامی نقاط اکسترمم در بالای خط روند رسم شده قرار بگیرند `up` خواهد بود.
- اگر تمامی نقاط اکسترمم در هر دستگاه در پایین خط روند رسم شده قرار بگیرد `down` خواهد بود و عدد 1 را چاپ کن. اگر تمامی نقاط اکسترمم در بالای خط روند رسم شده قرار بگیرند `up` خواهد بود و عدد 2 را چاپ کن.
- نقاط اکسترمم در هر دستگاه که خط روند از آن رسم شده است ماکزیمم هستند یا مینیمم؟

- $(\text{abs}(w_max) - \text{abs}(w_min)) / (\text{abs}(w_max) + \text{abs}(w_min)) * 200$
- اگر رابطه $(\text{abs}(w_max) - \text{abs}(w_min)) / (\text{abs}(w_max) + \text{abs}(w_min)) * 200$ برای هر دستگاه، بزرگتر و مساوی 50 بود عدد 1 را چاپ کن و در غیر این صورت عدد 0 را چاپ کن.

StabilityCoefficient_DSP: در این متد، DSP، ضریب پایداری قیمت، ضریب پایداری زمانی و نسبت این دو ضریب پایداری، محاسبه می‌شود.

Compute: محاسبات اصلی کلاس **InputData** در این متد انجام می‌شود که کاربر باید با استفاده از نمونه شی کلاس **InputData**، این متد را جهت اجرای برنامه فراخوانی کند.

TypicalPointExtremum: این کلاس برای محاسبه نقاط اکسترمم در n تعداد داده است. در این کلاس، از کلاس‌ها و متدهای ذیل استفاده می‌شود: **DataPoint**، **OperationsOnArrays**، **CONT**، **CONT_2** و **EMA**. در کلاس اصلی **InputData**، یک نمونه شی از کلاس **TypicalPointExtremum** ساخته می‌شود و سپس متد **EMA** فراخوانی می‌شود. **CONT**: این متد برای شمارش تعداد علامت‌های مثبت و منفی ستون‌های هیستوگرام مکدی در هر فاز مثبت و منفی است. **CONT_2**: این متد برای شمارش تعداد ستون‌های هیستوگرام مکدی در آخرین فاز مثبت یا منفی که در انتهای داده اتفاق می‌افتد، است.

EMA: در این متد، میانگین متحرک 12 و 26 روز و مکدی برای تمامی کندل‌ها محاسبه می‌شوند. نقاط اکسترمم تعیین می‌شوند. اولین کندلی که معین می‌کند در داده‌های گذشته یک اکسترمم وجود دارد نیز تعیین می‌شود.

OperationsOnArrays: این کلاس برای عملیات روی آرایه و لیست ایجاد شده است و شامل متدهای زیر است:

- **LineSpace**: برای افزایش یک بازه بکار می‌رود.
- **FindIndexArray**: پیدا کردن اندیس یک عضو مشخصی از آرایه.
- **FindIndexList**: پیدا کردن اندیس یک عضو مشخصی از لیست.
- **ConvertIntListToString**: تبدیل یک لیستی از داده‌های **Int** به **String**.
- **ConvertDoubleListToString**: تبدیل یک لیستی از داده‌های **Double** به **String**.
- **ConvertIntListToStringList**: تبدیل یک لیستی از داده‌های **Int** به لیستی از **String**.
- **ConvertDoubleListToStringList**: تبدیل یک لیستی از داده‌های **Double** به لیستی از **String**.
- **ConvertDoubleArrayToStringList**: تبدیل یک آرایه‌ای از داده‌های **Double** به لیستی از **String**.
- **ConvertIntArrayToStringList**: تبدیل یک آرایه‌ای از داده‌های **Int** به لیستی از **String**.

PositionPointsAnswer: موجودیت 7 نقطه اکسترمم متوالی که برای حل دو دستگاه متوالی استفاده می‌شود، تعریف می‌کند و این اطلاعات شامل موارد ذیل است: زمان پیوسته، زمان گسسته، قیمت در نقطه اکسترمم، زمان شکار که بیانگر وجود اکسترمم در اطلاعات قبلی است، فاصله تغییر فاز بعد تعیین اکسترمم اول با اولین زمان شکار جهت یافتن اکسترمم دوم، فاصله اکسترمم اول با اولین زمان شکار جهت یافتن اکسترمم دوم، زمان تغییر فاز بعد تعیین اکسترمم اول و جست و جو برای اکسترمم دوم.

Regression: این کلاس برای محاسبه ضرایب a_0 و a_1 در خط رگرسیون $r = a_1 t + a_0$ است که محاسبه آن توسط متد **ComputeRegr** در این کلاس انجام می‌شود. در این کلاس از کلاس **MathLine** برای ذخیره اطلاعات استفاده می‌کنیم.

MathLine: این کلاس، موجودیت یک خط را بیان می‌کند که به دو صورت ضمنی و صریح، معادله خط تعریف می‌شود.

DataTag: در این کلاس موجودیت روند هر دستگاه مشخص می‌شود که متد **Tags** در کلاس **InputData** از نوع این کلاس است.

UnknownAnswers: این کلاس، موجودیت ضرایب مجهول معادله زیر را مشخص می‌کند که با اجرای برنامه، این ضرایب بدست می‌آیند و به کاربر تحویل داده می‌شوند:

$$y = a_1 \sin(w_1 t + \theta_1) + a_2 \sin(w_2 t + \theta_2).$$

ضرایب مجهول عبارتند از: $a_1, w_1, \theta_1, a_2, w_2, \theta_2$.

AllTags: موجودیت تگ‌های مختلف را بیان می‌کند.

FinalResult: این کلاس، نتایج حاصله از اجرای برنامه را به کمک کلاس **ExcelSupport** و ایجاد نمونه شی از این کلاس و فراخوانی متد **Set_One_Cell** از کلاس **ExcelSupport**، در هر سلول اکسل نوشته و به کاربر تحویل می‌دهد.

Theta2: در این کلاس ضریب θ_2 محاسبه می‌شود. نکته‌ای که وجود دارد این است که محاسبات این ضرایب به هم وابسته هستند لذا به این دلیل بحث وراثت پیش می‌آید. ضرایب معادله فوق به ترتیب به صورت زیر بدست می‌آیند:

(1) a_1 و w_1

(2) θ_1

(3) a_2 و w_2

(4) θ_2

پس به کلاس‌هایی نیاز داریم که از هم ارث‌بری کنند. این ارث‌بری به صورت زیر است:

- 1) **A1W1 : Solver**
- 2) **A1W1_MiddlePoints : A1W1**
- 3) **Theta1 : A1W1_MiddlePoints**
- 4) **A2W2 : Theta1**
- 5) **Theta2 : A2W2**

کلاس فرزند	کلاس پدر	ارث‌بری
A1W1	Solver	A1W1 : Solver
A1W1_MiddlePoints	A1W1	A1W1_MiddlePoints : A1W1
Theta1	A1W1_MiddlePoints	Theta1 : A1W1_MiddlePoints
A2W2	Theta1	A2W2 : Theta1
Theta2	A2W2	Theta2 : A2W2

نکته: زمانی که بحث وراثت داریم، برای شی‌سازی، درونی‌ترین فرزند را صدا می‌زنیم و برنامه وقتی اجرا می‌شود در ابتدا سراغ بیرونی‌ترین پدر می‌رود.

نحوه بکارگیری کلاس‌ها برای محاسبه ضرایب معادله فوق به صورت زیر است:



Solver: این کلاس، یک کلاس پدر است و سایر کلاس‌های فرزند، ویژگی‌های این کلاس را ارث‌بری می‌کنند. در این کلاس، دو متد **ComputeRegressionLine** و **ComputeMiddlePoints** تعریف شده است.

ComputeMiddlePoints: این متد برای محاسبه نقاط میانی هر 6 نقطه اکسترمم متوالی است.

ComputeRegressionLine: این متد یک آرایه از نوع double خط رگرسیون با شیب a_1 و عرض از مبدأ a_0 ایجاد می‌کند (یعنی $r = a_1 t + a_0$).

A1W1: این کلاس، از کلاس **Solver** ارث‌بری می‌کند و خودش کلاس پدر برای کلاس **A1W1_MiddlePoints** است. چهار متد **F1**, **F2**, **YF1** و **CompA1W1** در این کلاس تعریف شده‌اند. متدهای **F1**, **F2** و **YF1**، تابع‌های مورد نیاز برای محاسبه w_1 و a_1 هستند. متد **CompA1W1** برای محاسبه w_1 و a_1 است. در این کلاس، از کلاس **OperationsOnArrays** برای افرازبندی یک بازه و پیدا کردن اندیس یک عنصر مشخصی از لیست، استفاده می‌شود. چون متد **CompA1W1** توسط کلاس فرزند **A1W1_MiddlePoints** ارث‌بری می‌شود آن را به صورت **virtual** تعریف می‌کنیم.

A1W1_MiddlePoints: این کلاس از کلاس **A1W1** ارث‌بری می‌کند و خودش کلاس پدر برای کلاس **Theta1** است. در این کلاس، دو متد **CompA1W1** و **ComputeA1W1** تعریف می‌شوند که متد **CompA1W1** از کلاس **A1W1** ارث برده است به این دلیل آن را به صورت **override** تعریف کرده‌ایم. از این دو متد از کلاس **A1W1_MiddlePoints**

متد **ComputeA1W1** باید در کلاس فرزند **Theta1** با استفاده از دستور **base** فراخوانی شود. متد **ComputeA1W1** برای یافتن λ بهینه، مدام متد **CompA1W1** را برای 5 نقطه میانی بکار می‌گیرد و اگر a_1 و w_1 بدست آمده مناسب نبود سراغ متد **CompA1W1** از کلاس پدر **A1W1** برای 6 نقطه می‌رود.

در این کلاس، از کلاس **OperationsOnArrays** برای افزایش یک بازه و پیدا کردن اندیس یک عنصر مشخصی از لیست، استفاده می‌شود.

Theta1: این کلاس از کلاس **A1W1_MiddlePoints** ارث‌بری می‌کند و خودش کلاس پدر برای کلاس **A2W2** است. در این کلاس، دو متد **F3** و **CompTheta1** تعریف می‌شوند. **F3** تابع مورد نیاز برای محاسبه θ_1 است. متد **CompTheta1** چون توسط کلاس فرزند **Theta2** ارث‌بری می‌شود آن را به صورت **virtual** تعریف می‌کنیم.

در این کلاس، از کلاس **OperationsOnArrays** برای افزایش یک بازه و پیدا کردن اندیس یک عنصر مشخصی از لیست، استفاده می‌شود.

A2W2: این کلاس، از کلاس **Theta1** ارث‌بری می‌کند و خودش کلاس پدر برای کلاس **Theta2** است. چهار متد **F4**، **CompA2W2**، **YF2** و **ComputeA2W2** در این کلاس تعریف شده‌اند. متدهای **F4** و **YF2**، تابع‌های مورد نیاز برای محاسبه a_2 و w_2 هستند. متد **CompA2W2** برای محاسبه a_2 و w_2 است. متد **ComputeA2W2** برای یافتن ضریب مناسب جهت تعیین بازه برای w_2 ، بارها اجرا می‌شود تا این ضریب بهینه پیدا شود. متد **ComputeA2W2** باید در کلاس فرزند **Theta2** با استفاده از دستور **base** فراخوانی شود.

در این کلاس، از کلاس **OperationsOnArrays** برای افزایش یک بازه و پیدا کردن اندیس یک عنصر مشخصی از لیست، استفاده می‌شود.

Theta2: این کلاس از کلاس **A2W2** ارث‌بری می‌کند. در این کلاس، دو متد **F5** و **CompTheta1** تعریف می‌شوند. **F3** تابع مورد نیاز برای محاسبه θ_2 است. متد **CompTheta1** چون از کلاس پدر **Theta1** ارث‌بری می‌کند آن را به صورت **override** تعریف می‌کنیم.

در این کلاس، از کلاس **OperationsOnArrays** برای افزایش یک بازه و پیدا کردن اندیس یک عنصر مشخصی از لیست، استفاده می‌شود.

الگوریتم برنامه به صورت زیر است:

1. تحویل اطلاعات از کاربر به برنامه‌نویس.
2. تعیین نقاط اکسترمم.
3. تعیین شیب و عرض از مبدأ خط رگرسیون.
4. محاسبه ضرایب معادله.
5. پیاده‌سازی تگ‌ها روی دستگاه معادلات.