Что сделано:

1. Инициализирован: class Angles, class Polygon, class DB (DataBase)
   1. Angles для хранения углов, для операций над углами (перевод из DD в DMS и обратно, контроль углов, которые измерялись на пунктах) и между ними, складывание, вычитание
   2. Polygon для хранения всех измерений, полученных на пунктах полигона. Внутри выполняются вычисления теории, исправленные значения и получение всей таблицы координат
   3. DB для чтения исходных данных с локального компа (заранее подготовленные данные), либо данных, которые будут передаваться через API, либо документом, либо через скрапинг указанных данных на вебе.
      1. В этот класс должен попадать JSON уже. Если что-то другое будет, надо будет проверять, что получает экземпляр класса и на основании проверки принимать решение, как читать, что проскрапить

Процедура вычисления таблицы координат

1. Указать измеренные горизонтальные углы и горизонтальное проложение на пунктах
2. Указать исходные координаты (координаты первого «твёрдого» пункта и последнего)
3. Указать какой дирекционный угол между ними
4. Посчитать практическую и теоретическую сумму углов
5. Посчитать невязку горизонтальных углов
   1. Посчитать теоретическую невязку горизонтальных углов (Зависит от выбранного прибора, по-моему двойная точность прибора или что-то такое)
6. Посчитать невязку, раскидать поправки в углы
7. Сверить исправленные горизонтальные углы с теорией ✔️
8. Инициализировать угол - class Angles
9. Инициализировать полигон – Polygon
   1. Передаю аргументы:
      1. from\_local - либо заготовленные данные в Полигон передаваться будут, либо сам пользователь загрузит (True, если заготовленные данные) и надо будет ещё номер (порядковый номер) этих данных прокинуть. На фронте типа появится доп. Окошко, в которое надо будет прокидывать количество допускаемых данных.
      2. data - Тут либо сразу данные (словарик/json) прокинутся, либо просто порядковый номер исходных данных из «бд»
10. Инициализировал объект, которым буду читать исходные данные - DB (внутри Полигона)

Может и записывать (сохранять тоже стоит через экземпляр DB)?

1. Работаю в полигоне
   1. Прочитываю все данные - get\_all\_data()
   2. Внутри Полигона разбиваю инфу на углы, на дир. Угол, на координаты исходные, и получу «вспомогательную сторону» (для информации, по какой формуле вычислять теоретическую сумму углов)
   3. Формирую список всех углов для дальнейшей обработки:
      1. Вычисляю теоретическую сумму горизонтальных углов - theoretical\_sum
      2. Вычисляю практическую сумму горизонтальных углов
      3. Вычисляю невязку
         1. После этого сделаю проверку: невязку разделю на количество углов (получу поправку, которую нужно внести в каждый угол) и сравню с одной секундой… Если полученная поправка будет равна меньше 1” значит надо раскидать по одной секунде, пока невязка не будет равна нулю во все углы на основании какого-то алгоритма, иначе раскидать невязку поровну во все углы.

Алгоритм: возьму список углов, отсортирую по убыванию и просто буду закидывать секунду в каждый угол в цикле while и пересчитывать DMS

* + - 1. После раскидки поправок надо будет написать функцию, которая снова возьмёт, проверит сумму углов через calc\_sum\_of\_practice\_angles и сравнит с теорией. Сама суть, что мне надо выполнить сравнение суммы в DD, но DD надо получать уже из ИСПРАВЛЕННЫХ DMS. Т.е. я сначала исправляю, используя только DD, потом при исправлении я делаю пересчёт DMS и потом при проверке я проверяю через новый DD (надо это через сеттер автоматизировать, если изменяю что-то в угле, то надо делать пересчёт DMS и после сразу пересчёт DD, что б при сравнениях/вычислениях не использовать метод экземпляра, а использовать атрибут DD у экземпляра.
    1. Вычисляю теоретическую невязку (в зависимости от прибора, от его точности, двойная точность прибора, всякое такое)
    2. Вычисляю поправку, которую нужно внести в каждый угол

НАДО КАК-ТО НАУЧИТЬ класс Angle если я передаю в DMS, то один вариант инициализации, если DD, то другой вариант инициализации. Например, буду вычислять сумму углов или ещё что-нибудь, а мне надо вычислять будет потом DMS и т.п.

Либо передавать если передаю массив аргументов, то по ним надо пройтись и раскидать в нужные атрибуты, а если одно значение передаю, то это DD

# Надо бы

Написать класс для взаимодействия с АПИ

Там должно быть что-то типа как класс DB, методы на получение и на отдачу данных, их буду использовать внутри класса Полигон, также как с БД

Либо просто модифицировать модуль БД

Класс Angle1:

Добавить методы для того, чтобы можно было отнимать и прибавлять углы, например. Прибавление есть

Подумать над тем, чтобы хранить там тоооолько углы и всё, а со сторонами что-то придумать... как-то не очень, что у каких=то углов расстояние нулю равно у кого-то оно есть и всякое такое.

«Т.е. я сначала исправляю, используя только DD, потом при исправлении я делаю пересчёт DMS и потом при проверке я проверяю через новый DD (надо это через сеттер автоматизировать, если изменяю что-то в угле, то надо делать пересчёт DMS и после сразу пересчёт DD, что б при сравнениях/вычислениях не использовать метод экземпляра, а использовать атрибут DD у экземпляра.»

Придумать то, чтобы при получении суммы углов, например с исходными данными №3 не было такого, что принтуется угол 2339°59'60", хотя угол 2340 в DD такой пишется. Как-то накидать проперти для гет и сета, и чтобы шла проверка, если больше какой-то величины получатся – то надо закинуть в следующий разряд.

Призадуматься, для получения суммы периметра нормально использовать этот же класс angle1? Либо надо наследование какое-то делать, сначала угол, а потом класс угол+длина линии, чтобы в первом классе можно было считать только величину угла, и она может быть бесконечна. НО при использовании класса угол+длина линии, там должно быть ограничение в 360 градусов, иначе надо делать – или + 360 градусов.

# ВОЗМОЖНО

Класс Polygon

должен представлять из себя связанный список (одно или двусвязный, без разницы

Что б точно не проебаться в том плане, какой угол за каким следует, что б каждый пункт знал, какой за ним или до него следует. Нужно ли?

Что б случайно при какой-нибудь сортировке не напутать.

Надо сделать отдельный класс для хранения всех объектов, которые находятся в строке…. Иначе у меня есть класс angle, я туда добавил длину стороны… Но разве туда надо пихать дирекционный угол, приращения координат, координаты?

Надо сделать отдельный класс поинт… Возможно это будет связный список и будет всё зоебись

# На тестирование или на подумать

Почему при невязке 0,0,-5 первое условие раскидки не отрабатывало, оно не отнимало ни у какого угла секунду??

Потестить с другими данными