🧭 Тестовое задание: расчёт и визуализация риска столкновения судов (CPA / TCPA)

📌 Цель

Создать небольшое приложение, которое анализирует движение судов по данным радара и определяет, есть ли риск столкновения.
Задание имитирует ключевую часть системы EasyCaptain — *Decision Engine* для предотвращения столкновений.

⚙️ Описание задачи

Представь, что наш катамаран движется в море, а радар обнаруживает рядом другие суда.
Для каждого объекта известны:

- координаты (x, y) в метрах относительно нашего судна,
- курс (в градусах),
- скорость (в м/с).

Твоя задача:

1. Вычислить для каждого объекта **CPA (Closest Point of Approach)** — минимальное расстояние до столкновения;
2. Вычислить **TCPA (Time to CPA)** — время до этого момента;
3. Определить, есть ли **риск столкновения**, если CPA < 50 м и TCPA < 30 секунд;
4. Отобразить результат визуально (графически или текстом).

🧮 Формула (подсказка)

Для упрощения задачи можно использовать стандартные векторные расчёты CPA/TCPA между двумя движущимися объектами:

r = P_target - P_ownship (вектор расстояния)
v = V_target - V_ownship (вектор относительной скорости)
tcpa = - (r · v) / |v|²
cpa_distance = |r + v * tcpa|

Если tcpa < 0, объекты расходятся.

💻 Требования к реализации

Вариант A — Python

- Программа читает данные из CSV (пример ниже).
- Реализует расчёт CPA / TCPA.
- Показывает результат в консоли и визуализирует сцену с помощью matplotlib.
- Визуализация: синяя точка — наш катамаран, красные — другие суда, зелёным выделяются опасные объекты.

Пример CSV:

id,x,y,speed,course

1,100,50,5,180
2,-120,30,3,90
3,200,-100,8,210

Пример вывода:

Target 1: CPA=42.3m, TCPA=12.5s ⚠️ COLLISION RISK
Target 2: CPA=85.1m, TCPA=47.2s ✅ Safe
Target 3: CPA=65.7m, TCPA=33.0s ✅ Safe

Вариант B — C++

- Реализовать то же самое с чтением данных из CSV или ручного массива.
- Выводить таблицу результатов в консоль.
- Можно добавить простую **ASCII-графику** — «радар» с точками (O — наш катамаран, x — другие суда).

🎨 (Необязательно, но приветствуется)

- Реализовать **фильтр Калмана** для сглаживания координат.
- Сохранять результаты в JSON.
- Добавить REST API (/alerts) с выводом списка опасных объектов (если знаком с FastAPI, cpp-httplib и т.д.).

📦 Что отправить

1. Исходный код (Python или C++).
2. Короткий README (1 страница):
    - как запускать программу,
    - что реализовано,
    - пример вывода или скриншот визуализации.
3. (опционально) описание, как этот код может использоваться на Jetson или в бортовой системе.

⏱️ Оценка и сроки

⏰ Время выполнения: до **5 дней** с момента получения.
🔍 Мы оцениваем:

- чистоту и читаемость кода,
- инженерную логику,
- точность расчётов,
- качество визуализации,
- понимание физики задачи.

🚀 Контекст проекта

EasyCaptain — интеллектуальный *AI co-pilot* для катамаранов, который подключается к радарам и автопилоту, анализирует обстановку и помогает избегать столкновений.
Этот тест — маленький кусочек реального Decision Engine, который мы разрабатываем.