Task Description :

The dummy data file provided with the work package provides you a time series data in json format where each entry is a single sensor(camera) with what it has detected and their metadata and the second file is .csv file which has heading angle, state (stationary/driving) and acceleration of one vehicle(forklift). Each sensor has a **timestamp**, **object_classes**, **object_positions_x_y**, along with other information which is not relevant for the task. It is a data of a scenario where some objects are moving around and detected by a sensor network. Each object might be detected by more than one sensor.

Task is to write an algorithm that reads the data sequentially from both files(one by one) according to timestamp and cluster the data coming from different sensors (use "cam_<id>") and save the fused data to a new csv file where each entry should have f_timestamp, f_id, cluster_data, heading, status. The data in the new csv file must be updated sequentially and should not be done all at once.

You have to follow following rules/constraints while doing the above task:

1. **object_positions_x_y** for the same object from 2 or more different sensors at the same timestamp might be different.

2. **object_ids** is specific to individual sensor only and has no relation to the id from another sensor. The sensor assigns a random ID to each detected object. At same timestamp values, if one sensor is showing 2 or more different objects with different id then these objects are different for sure

3. **timestamp** format is in YYYY-MM-DD HH-MM-SS.003. This should be the key defining factor for fusing data from different sensors.

4. **Clustering** of data should be performed based on the positions (x, y) and distance metric but at the same time keeping in mind the rule number 1,2 and 3. The maximum distance threshold to fuse any data together is 2 meters.

5. **cluster_data** is a list of lists, where the the individual list inside the main list contains data in the format [x_position, y_position, sensor_id]. These values are from the first json file that are clustered together based on above rules.

6. **f_id** should be randomly assigned to the clustered data and this f_id should remain same for the same clustered object

7. **f_timestamp** is the value of a common timestamp from the entries that were clustered together. You can use averaging of timestamp wherever required

8. **heading** is the filtered heading angle which can be exracted from the imu data sheet. In the csv file the heading values are noisy, use any **kalman filter** to smoothen the heading angle such that the heading direction of the vehicle is smooth.

9. **status** is the status of the vehicle which can be extracted from the imu data sheet

10. Reduce the time complexity of the algorithm to as minimum as possible. Time of execution of code will also be taken into account by us.

What should you do:

1. **Understand the data.**
2. **Read and parse the json data and csv data using and library you want (sequentially using the timestamp information available).**
3. **Write a fusion algorithm where you compare data from different sensors and based on the rules above you cluster objects / fuse objects from different sensors.**
4. **Save the results to a new csv file named "fused_data.csv" as mentioned in the introduction of the task**

The given task must be completed in the mentioned time frame. You have the first 1 hour minutes to understand the task and ask questions related to the task. After 1 hour, questions will not be answered and you can assume anything which is missing or which might not seem correct to you.

A faster time of submissions will be considered during the evaluation.

You can also submit incomplete task. It is not necessary to submit complete task. A comparative evaluation is done and in past candidates with partial results have also been accepted for next round of evaluation.

The final format of submission should be a private GIT repo which you should share with a.chowdhury@sentics.de

A video file is also required where you present the solution while executing the whole code and provide explanation regarding what you have done. Share this video with me at a.chowdhury@sentics.de

Code should be written using **C++ only. You are allowed to use any library of your choice.