FAZUA Embedded System Prodigy

Exercise 1: Create a classifier for checkbase

In the production line of the drivepack (the module where the motor is integrated), there are 2 machines for testing. One of them is in the middle of the assembly line (precheck) and the second one is at the end (End Of Line).

The precheck tool is a binary classifier, using 2 type of sensors, it will decide if the motor is “OK”, ready to continue with the assembly or “fail” , in this case, the motor will be rejected and will be moved out from production line.

All the motors that has an “OK” in the precheck tool, will be fully assembly and tested in the EOL.

The EOL tool is also a binary classifier, using several variables (that are not need it), makes a decision if the motor is ready to sell “pass” or if it is not “fail”.

We want to improve the checkbase so the result of it is as close as possible as the EOL result. In an ideal situation, all the motors that has a “OK” in the checkbase would “pass” the EOL. (And all the motors that “fail” the checkbase would “fail” the EOL).

Task 1:
Create a binary classifier modeled with the data form “input 1” and /or “input 2” so the output (pass 1, fail 0) would be close to the EOL result . You can use logistic regression, random forest, neural networks... or just using a threshold in the values

Task 2:
Adjust the classifier to reduce the “False negative**” ratio less than 10%.

**False negative: is consider a false negative a motor that will be a “fail” in the checkbase test and a “pass” in the End Of Line.

You can use an excel file to solve the problem. Basic knowledge of excel is enough. There is no need to use very special functions. (I always work with google sheets)

Exercise 2: Create an agenda for my grandmother

My grandmother has a very bad memory. She has Alzheimer and always forgets what she has done and also what she has to do. Unfortunately I live in the city and cannot help her, but I believe that creating a scheduling program can help her a lot in her daily life.

She needs a tool that works as an interactive agenda. This interactive agenda will answer you with the activity you are supposed to be doing in an exact time. It will also advertise you when an scheduled activity is starting and also 10 minutes before it finish (not in all the cases)

The scheduled activities have an internal state: done and undone. All the activities are in "undone" state in the beginning.

When you write down a time, the agenda is answering with the activity you are supposed to do. If the activity is not done, then the agenda should ask if you are doing it. If you answer "yes" then the agenda should change the internal state from "undone" to "done. If you answer "no" then the internal state should not change.
There should be a 3 second interval time between agenda printed outputs.

The program should keep in a silent state, there are 3 cases that the program should create an output (some printed words):

- After the user writes a time (to ask what to do).
- When a scheduled task just starts.
- 10 minutes before a scheduled task finish (only if the task is "undone")

Example 1: If I enter "8:35" in the program the answer would be "Breakfast time". In case that the program has breakfast as an undone task, the program would ask after 3 seconds "Are you having breakfast?" and my grandma would answer "yes" if she is having, so the state of the breakfast time would change to "done". If the answer is "no" then, the state of breakfast time would keep in "undone".

Example 2: if I enter "now" and the task is "having lunch" and the task is already done, then the program should answer something like "Chill, you already have lunch".

Task 1:
Draw a flowchart with the agenda specifications.
Create a C program according to this flowchart. It should be able to read the input (through terminal). Using printf for output is enough, no graphical interface is needed.
I don't want my grandma to get bored, so I would like you to create a schedule with at least 8 activities perfectly scheduled in a 0-24h clock.

Task 2:

I am not my grandmother, so if you can add a way to run the day faster. Adding a speed factor when executing the program (speed factor is =1, time run normally, speed =2 double speed…)
This is just the solution I found to a problem. If you think you have one better, please programm it. Also, new useful features are welcome.

The code must be done in a clear, **organic** and **modular way**, even if it is a small project.

You will send me the source code file/es. **I will compile, execute them in my linux termina**l. You can send it to me by email, uploaded to a GIT repository or add it to a blockchain.

Good luck!