**Data logger software**… logging values are very easy .. and done

**research on algorithm quantisation**

My experiences

While writing code is important, documentation of everything is equally important. Also writing experiences at regular intervals of time, for an hour say, is very beneficial.

**Documentation**

The documentation from the datasheet must be done using screenshots. Even if the datasheet is very concise, taking screenshots and writing some specks in them is very beneficial and easy to understand.

The documentation must also be made and managed in a revision control as well.

**Code**

The code has to necessarily written in the form of revision control system. This is very important especially when continuous improvement or fixing of bugs is done over a time span. When the project has to undergo some more than 10 revisions then the revision control system is very necessary.

The patch system should be chosen and entries should be done about the changes that fix the bugs or introduces some improvement in software.

The main loop should be small, up to the level that everything is clearly visible in the loop itself without the need of scrolling. Although time taken to switch between functions should be considered and noted in the particular platform in which we intended to work in.

Not only the main loop, but the whole code should be properly modularized so the readability of code can be enhanced and bug detection and removal can be easily done.

**Code representations**

Code has to be properly represented in flowcharts, algorithms and state diagrams for each revision of the software. So either, get some software that will map the flowcharts in the computer for each separate revision or get this in a separate notebook.

As identifying bugs and then fixing them is a great pain and take a lot of time in code writing, sometimes situation comes such that a person fears even to touch the code because he thinks some bug will get into it. A separate notebook should be maintained for each project code. This notebook should contain some important information about the bugs found in the software and some other sensitive code related issues.

**Team**

Coding has to be mentored or mastered by one person, so all the software maintenance lies in single hand but two to three people can collaborate in software bug removal and testing. That’s a big relief. People who want to contribute to code must submit patches as in Ubuntu, the master coder can choose to apply, reject or merge a particular patch.

After making one version of the code, the code should be send to the **testing team** so that it can test the code and find possible errors and bugs/improvements and documents the bugs in a report along with the result of applying these on the codes. Meanwhile the software writer can work on some more improvements in the code or some alternate code writing.

This way one person can convert the brainstorming into code and possible practical solutions while the testing team will test each version of the code and in the last the best possible solution can be chosen for the finals.

**At competition**

During the competition, we should try our most optimal and the most precise code at first. This will build the required confidence and so we can later try more speed in the later tries. Also we should focus on testing there and no code without testing should be introduced in the final software.

Also another important thing is that we should not add some new feature in the code on the competition day. If some very important need is there then there should be parallel testing going on by the testing team.