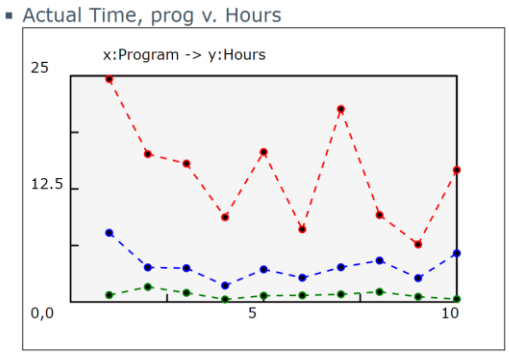
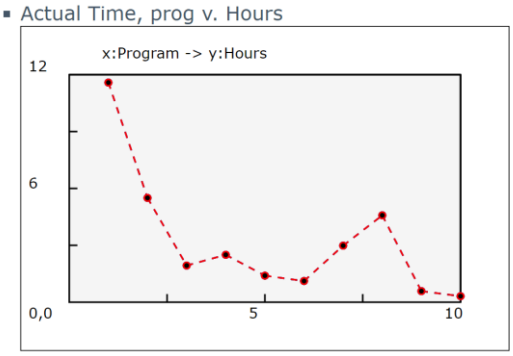
Analysis of Time Estimates.

The time estimates is mainly use for measure the effort that use in the program and make it possible to make it predictable for developer to manage their time spend on the software project.

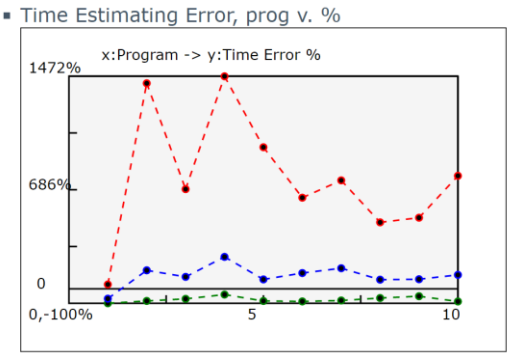
Here is the actual time the class and I use in the ten program.

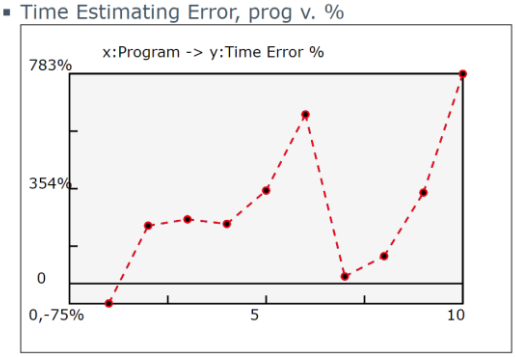




Class Myself

And here is the time estimation error.





According to the diagram, we can see that the time estimation what I did is much above the 70% when it comes to program06, program09 and program10. But other program time estimate is still in 70%. The accurate of time estimating is not improved during the process, seem that I can finished the program in a really short time if I familiar with the algorithm.

As we can see in the diagram, program06, program09 and program10 is the main part I am not in the 70% prediction part. After I review these program I found out that the program06 is a simple program I didn't realize before I come to it. It only solve the relationship between x and p that I have already done in program05 but just reversed. I can just use program05's function to do the majority thing, so I just import program05 and that save a lot of time and cause the time estimate error raise to an unbelievable level. To program10, it is the same reason, the main sort function is not changed, program10 is just a program based on program09 and add some arguments and change the comparison method according to the arguments. Program09 I am doing so quick is that because I am so familiar with quick sort, the first time I see the O(nlogn) I know that this will be the heap or quick sort. And I use the most familiar sort method I know and write it so quickly

without thinking and that make the time estimate error so high.