Being a more effective programmer

Tom Rochette <tom.rochette@coreteks.org>

November 2, 2024 — 36c8eb68

0.1 Context

Programmers that like their craft want to improve as much as they can. However it becomes difficult to determine how one can improve given the variety of paths possible. This article tries to examine what are considered best practices and ways to be as efficient as possible in your work.

0.2 Learned in this study

0.3 Things to explore

- How to more efficiently develop complex algorithms
- How to be more careful when writing new code
- How to spend less time debugging
- Is writing a working solution better than thinking about a better alternative solution to develop straight away?
- How to recognize where to look for the source of a problem/bug?
- How can you make reusing your own code more easy?
  - How can you find code you wrote before rapidly?

1 Objectives

- Minimize the time spent understanding requirements
- Minimize the time spent developing a feature
- Minimize the time spent debugging an issue

2 Basics

- Use a consistent code style
- Spend time learning the language features available functions/classes

3 General

- Go through code you have issue writing yourself with someone else to see how they reason about it
- Deliberately practice the aspects of programming you have difficulty with
  - Practice going through code challenges not for the purpose of solving the problem, but observing how you solve the problem and where you could improve
- Do not optimize for speed, optimize for readability/understandability/transparency

4 Effective code writing

4.1 Before

- Understand the problem
- Understand the requirements of your problems
- Understand the time and space complexity requirements

4.2 During

- Write test cases
- Execute your test cases often
- Make explicit (write/explain) the important parts/steps of the algorithms
- Pseudo-code your solution, then implement it
- Break the problem into parts and solve the parts
- Take a step back from time to time to observe your existing solution and attempt to determine if it is too complex for what it is trying to solve
- Simulate program execution

5 Effective debugging

- Check the problem definition
- Run the program step by step using the debugger and think about the (desired) state at each point
-

6 Effective issue fixing

- First, understand the reported issue
-

6.1 2018-05-12

- Start with the simplest cases you want to support, then proceed to more and more difficult cases
- Verify your progress in small iterations to prevent writing a lot and then having to make it all work
- Identify the variables that you need through an iterative process
- Write comments about the alternative options you have
- You must be willing to accept that some of your assumption may be wrong: Sometimes you will discover that what you took for granted is not necessarily correct (such as template code provided to you that does not accomplish what it states it does)

6.2 2018-05-18

- Record yourself solving a problem then take some time to analyze and determine the steps of the problem you were trying to solve. Determine how long you would expect each step to take and where there is potential for improvement
- Design a process for designing, implementing and debugging programs and follow it, changing it as needed

6.3 2018-05-21

- Define case for empty values as inputs

7 See also

8 References

- http://programmer.97things.oreilly.com/wiki/index.php/Coding_with_Reason
- http://programmer.97things.oreilly.com/wiki/index.php/Do_Lots_of_Deliberate_Practice
- https://softwareengineering.stackexchange.com/questions/65705/how-to-code-faster-without-sacrificing-quality
- https://www.quora.com/How-do-I-train-myself-to-code-faster-and-with-fewer-bugs
- https://news.ycombinator.com/item?id=9242245
- https://news.ycombinator.com/item?id=14708350
- http://arnklint.com/technology/become-efficient-programmer.html
- https://softwareengineering.stackexchange.com/questions/44177/what-is-the-single-most-effective-thing-you-did-to-improve-your-programming-skil
- https://medium.freecodecamp.org/how-to-think-like-a-programmer-lessons-in-problem-solving-d1d8bf1de7d2
- 7 Steps to Solve Algorithm Problems

8.1 Productivity

- https://www.lesswrong.com/posts/JTHe5oGvdj6T73o4o/how-i-am-productive