What metrics would be usable to determine expertise level in a particular programming language

Asked 15 years, 10 months ago Modified 14 years, 5 months ago

Viewed 3k times

6

I am interesting in the raw (or composite) metrics used to get a handle on how well a person can program in a particular language.

Scenario: George knows a few programming languages and wants to learn "foobar", but He would like to know when he has a reasonable amount of experience in "foobar".

I am really interesting in something broader than just the LOC (lines of code) metric.

My hope for this question is to understand how engineers quantify the programming language experiences of others and if this can be mechanically measured.

Thanks in Advance!

metrics

Share

asked Feb 12, 2009 at 1:53

Improve this question

Follow

John Burley

1,208 ● 2 ● 15 ● 22

pluralsight has a whole section on that where you can take some tests with some tricky questions and depending on your score they give you an expertise level such as beginner,proficient or expert. Depending on the answers you gave they will then suggest you the areas you need to improve at, on the selected language/framework etc. That gives you a hint about the main areas you need to cover as a professional. – Nasia Makrygianni Jan 31, 2020 at 9:57

15 Answers

Sorted by: Highest score (default)

3

In reply to the previous two posters, I'd guess that there *is* a way to get a handle on how well a person can program in a particular language: you can test how well someone knows English, or Maths, or Music, or Medicine, or Fine Art, so what's so special about a programming language?

In reply to the OP, I guess the tests must assess:

- How well you can program
- How well you can use the programming language

Therefore the metrics might be:

1. What's the goodness of the person's programming (and there are various dimensions of goodness such as bug-free, maintainable, quick/cheap to write, runs quickly, meets user requirements, etc.)?

2. Does the person use appropriate/idiomatic features of the programming language in question in order to do that good programming?

It would be difficult to make the test 'mechanical', though: most exams that I know of are graded by a human examiner. In the case of programming, part of the test could be graded mechanically (i.e. "does it run?") but part of it ("is it understandable and idiomatic?") is intended to benefit, and is better judged by, other human programmers.

Share Improve this answer

Follow

answered Feb 12, 2009 at 2:15

ChrisW

56k ● 14 ● 120 ● 233

Yes, we all like to think programming is somehow "special" don't we! :) I agree that (1) programming ability can be measured well using a well-thought-out test, and that (2) there is probably no satisfactory "mechanical" way to do this testing. – j_random_hacker Feb 12, 2009 at 2:35

... however the problem with many "test questions" available on the net is that they are bad questions (e.g. "How do you swap two values without a temporary"). You ideally want open-ended questions that are graded by someone whose expertise in that language you trust. – j_random_hacker Feb 12, 2009 at 2:37

3

The best indicator of your expertise in a particular language, in my opinion, is how productive you are in it.

Productivity is not just how fast you can work but, importantly, how few bugs you create and how little refactoring/rework is required later on.

For example, if you took two languages you have similar level of experience with, and were (in parallel universes) to build the same system with both, I would say the language you build the system with faster and with fewer defects/design flaws, is the language you have more expertise in.

Sorry it's not a "hard" metric for you, it's a more practical approach.

Share Improve this answer Follow

edited Feb 12, 2009 at 2:26

answered Feb 12, 2009 at 2:15

Ash

62.1k ● 31 ● 155 ● 172

I agree that that is the most *important* measure, but unfortunately it's one of the most impractical to measure. – j_random_hacker Feb 12, 2009 at 2:40

It can be measured by observation I suppose. eg. Someone who applies for say an "expert C# programmer" position should be able to produce more relaible code more quickly than most other C# programmers. – Ash Feb 12, 2009 at 4:29

That's where an important distinction between natural and computer languages lies. I've programmed in C++ for five years and in Python for 2-3 months scarcely but I am sure I

can develop anything but drivers, much quicker in Python than in C++. Doesn't work that way with French or Russian. – mannicken Feb 15, 2009 at 16:58

" I would say the language you build the system with faster and with fewer defects/design flaws" -- the issue I have with this is that problems often lend themselves to certain languages, frameworks, or paradigms. I wouldn't implement a high-performance, low-level graphics library in Python, nor would I do a bunch of high-level, low volume natural language processing in C. – Adam Parkin Jan 17, 2015 at 0:19

2

I don't believe that this can be "mechanically measured". I've thought about this a lot though.

Share Improve this answer

Follow

answered Feb 12, 2009 at 2:01

12345

529 ● 5 ● 7

2

Hang on...

Even the "LOC" of a program is a heavily disputed topic! (Are we talking about the output of cat *.{h,c} | wc -l or some other mechnanism, for instance? What about blank lines? Comments? Are comments important? Is good code documented?)

Until you've realised how pointless a LOC comparison is, you've no hope of realising how pointless other metrics

are.

Share Improve this answer

Follow

answered Feb 12, 2009 at 2:44

Arafangion

11.9k ● 1 ● 43 ● 74

1

It's a rather qualitative thing that is rarely measured with any great accuracy. It's like asking "how smart was Einstein?". Certification is one (and a reasonably thorough) quantitative indicator, but even it falls drastically short of identifying "good programmers" as many recruiters discover.

What are you ultimately trying to achieve? General programming aptitude can be more important than language expertise in some situations.

If you are language-focussed, taking on a challenge like Project Euler using that language may be a way to track progress.

Share Improve this answer

Follow

edited Feb 12, 2009 at 2:30

Svante

51.4k ● 11 ● 83 ● 124

answered Feb 12, 2009 at 2:10

Rog

4,085 ● 2 ● 26 ● 35

How proficient they are in debugging complex problems in that language.

1

Ask them about projects they have worked on in the past, difficult problems they encountered and how they solved them. Ask them about debugging techniques they have used - you'll be surprised at what you'll hear, and you might even learn something new ;-)

A lot of places have a person or two who is a superstar in their field - the person everyone else goes to when they can't figure out what is wrong with their program. I'm guessing thats the person you are looking for :-)

Share Improve this answer

Follow

answered Feb 12, 2009 at 3:11

Vladimir

1,106 ● 1 ● 7 ● 4

1

Facility with a programming language is not enough. What is required is facility with a programming language in the context of a partiular suite of libraries on a particular platform

- C++ on winapi on Windows 32bit
- C++ on KDE on Linux
- C++ on Symbian on a Nokia S60 phone
- C# on MS .NET on Windows
- C# on Mono on Linux

Within such a context, the measures of competence using the target language on the target platform are as follows:

- The ability to express common patterns succinctly and robustly.
- The ability to debug common but subtle bugs like race conditions.

It would be possible to develope a suite of benchmark exercises for a programmer. One might also, once significant samples were available, determine the bell curve for ability. Preparing these things would take literally years and they would rapidly be obsoleted. This (and general tightness) is why organisations don't bother.

It would also be necessary to grade people in both "tool maker" and "tool user" modes. Tool makers are very different people with a much higher level of competence but they are often unsuited to monkey work, for which you really want a tool user.

Share Improve this answer

Follow

answered Feb 12, 2009 at 3:24

Peter Wone

18.7k ● 14 ● 94 ● 147

1

John

There are a couple of ways to approach your question:

1) If you are interviewing candidates for a particular position requiring a particular language, then the only measure to compare candidates is 'how long has this person been writing in this language.' It's not perfect - it's

not even very good - but it's reality. Unless you want to give the candidate a problem, a computer, and a compiler to test them on the spot there's no other measure. And then most programmer-types don't do well in "someone's watching you" scenarios.

2) I interpret your question to be more of 'when can I call MYSELF profecient in a language?' For this I would refer to levels of learning a non-native language: first level is you need to look up words/phrases in a dictionary (book) in order to say or understand anything; second level would be that you can understand hearing the language(or reading code) with only the occasional lookup in your trusted and now well-worn dictionary; third level you can now speak (or write code) with only the occasional lookup; fourth level is where you dream in the language; and the final levels is where fool native speakers into thinking that you're a native speaker also (in programming, other experts would think that you may have helped develop the language syntax).

Note that this doesn't help determine how good of a programmer you are - just like knowing English without having to look up words in the dictionary doesn't show "how gooder you is at writin' stuff" - that's subjective and has nothing to do with a particular language as people that are good at programming are good in any language you give them.

answered Feb 15, 2009 at 16:37

Les Grove

Les, I like your answer! You understood that I was looking for measures for myself ("George"). I think the metric you might be suggesting is how many times you need to look up something in a programming manual per program for either read or writing of code. This over time(programs) could be trended. – John Burley Feb 15, 2009 at 19:51

In this natural language context, I would put a level between dreaming and being mistaken for a native speaker: 4.5: you dislike using other languages due to the mismatch between the shape of their constructs and the way your thoughts are organised. – Peter Wone Feb 16, 2009 at 23:41

1

The phrase "a reasonable amount of experience" is dependent upon the language being considered and what that language can be used for.

A metric is the result of a measurement. Stevens (see wikipedia: Level Of Measurement) proposed that measurements use four different scale types: nominal (assigning a label), ordinal (assigning a ranking), interval (ordering the measurements) and ratio (having a non-arbitrary zero starting point). LOC is a ratio measurement. Although far from perfect, I think LOC is a relevant, objective number indicating how much experience you have in a language and can be compared to quantifiable values in the software industry. But, this begs the question: where do these industry values come from?

Personally, I would say that "George" will know that he has a reasonable amount of experience when he has

designed, implemented and tested a project, maybe of his own choosing on his personal time on his home computer if need be. For example: database, business application, web page, GUI test tool, etc.

From the hiring managers point of view, I would start off by asking the programmer how good s/he is in the language, but this is not a metric. I have always thought that the best way to measure a persons ability to write programs is to give the programmer several small programming problems that are thought-out in advance and solved in a given amount of time, say, 5 minutes each. I have never objected to this being done to me in job interviews. Several metrics are available: Was the programmer able to solve the problem (yes or no - nominal)? How much time did it take (number of minutes - ratio)? How effective was their approach to solving the problem (good, fair, poor - ordinal)? You learn not only the persons ability to write code, but can observe several subjective things as well, such as their behaviour as they go about solving the problem, the questions s/he asks while solving the problem, the ability to work under pressure, etc, From a "quality" perspective though, remember that people do not like being measured.

answered Feb 20, 2009 at 5:02

John Ewing

51 • 2

1

Still, I believe there are some good metrics like the McCabe Cyclomatic Metric for cyclomatic complexity or the amount of useful commentary per block of code or even the average amount of code written between two consecutive tests.

Share Improve this answer

Follow

answered Jul 6, 2010 at 14:03

Valentin Brasso

1,387 ● 7 ● 21 ● 42

0

I know of no such thing. I don't believe there's concensus on how to quantify experience or what "reasonable" means. Maybe I'll learn something too, but if I do it'll be a great surprise.

This may be pertinent.

Share Improve this answer

Follow

edited May 23, 2017 at 12:02

Community Bot

1 ● 1

answered Feb 12, 2009 at 1:59

duffymo

308k ● 46 ● 374 ● 565

0

I find that testing the ability to debug is a more accurate gauge of programming skill than any test aimed at straightforward programming problems that I have encountered. Given the source for a reasonably sized

class or function with a stated (or unstated, in some cases) misbehavior, can the testee locate the problem?

Share Improve this answer

Follow

answered Feb 12, 2009 at 2:19

Sparr

7,712 ● 36 ● 49

0

Well, they try that in job interviews. There's no metric, but you can assess a person's abilities through questioning and quizzing.

Share Improve this answer

Follow

answered Feb 12, 2009 at 2:31

Frank

66k ● 96 ● 241 ● 327

0

WTF/s * LOC, smaller is best.

Share Improve this answer

Follow

answered Feb 12, 2009 at 2:50

Edwin Jarvis

6,120 ● 7 ● 38 ● 41

0

there are none; expertise can only be judged subjectively relative to others, or tested on specifics (which has its own level of inaccuracy)

see what is the fascination with code metrics for more information

Share Improve this answer

Follow

edited May 23, 2017 at 9:57

Community Bot

1 ● 1

answered Feb 12, 2009 at 2:58

Steven A. Lowe

61.1k ● 19 ● 135 ● 204