Examination of the Effects of Programming Languages on Developer Productivity, as Measured by Time

Ben Raz <ben.raz@student.winstonprep.edu>

July 21, 2023

Abstract

In this study, we test whether programming languages have an effect on developer productivity and how that effect manifests itself by implementing a real-world algorithm in each language. We find that lightweight scripting languages like Python and JavaScript give developers the most productivity while low-level languages like Rust and especially C++ do more to hinder developer productivity. Additionally, we summarize the current literature and discuss the limitations of this study.

1 Introduction

In one year on earth, developers will spend around 7 billion hours[1] interacting with a programming language. That is comparable to the lifetime of planets. At this scale, productivity clearly matters. A change of as little as 1% can free up tens of millions of hours. These hours can go towards fueling innovation. Because of the importance of programmer productivity, we study which popular programming language makes developers the most productive. Additionally, we analyze the tradeoffs of each language. According to both a survey of language popularity[3] and a study analzying the ease of use of writing a string parsing function in a number of different programming languages[2], Python and other scripting languages are prefereable in terms of developer efficiency.

2 Experimental Design

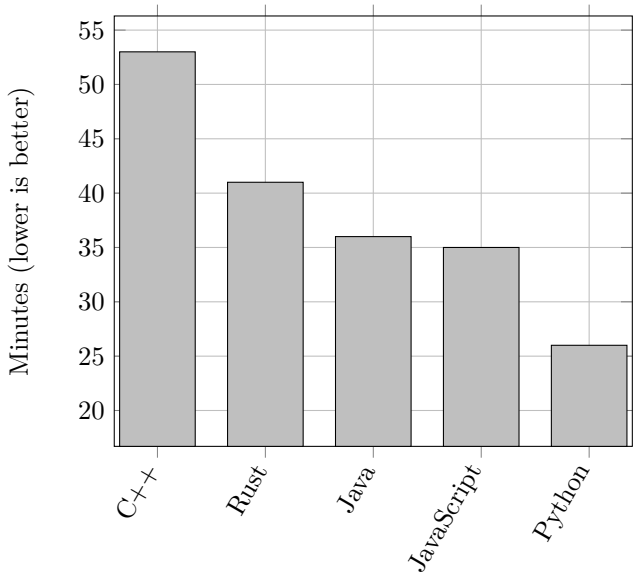
The experiment consisted of making a program that organizes files in a directory into sub-folders based on their file type. For example, it would take `image.png` and change it into `Png/image.png`. This task requires many things. First, it requires a good standard library as well as robust error handling and string manipulation. It is also similar to real-world work, which is very focused on the filesystem. This program is to be implemented in 5 different languages:

- C++
- Python
- JavaScript
- Rust
- Java

Once the development environemnt is set up, a stopwatch is started while the software is being written. If, for any reason, a break must occur, the stopwatch is paused and then unpaused as neccessary.Finally, the times are written down in a notes document once development is over. This process repeats for each programming language.

3 Results

The results for each programming language are tabulated below.



According to these results, A winner is clear. Python took only 25 minutes while JavaScript took nearly ten minutes more. Java took 36 minutes while Rust took 41. The C++ implementation took 42 minutes to write, more than twice the time took to write the python implementation.

4 Limitations of This Study

While our findings to corroborate similar studies on the topic, we acknowledge the limitations of this study. Firstly, the results may have been distorted by the fact that, for a given person, profficiency varies by the programming language. By only having one person undergo the experiment, we risk generalizing the varied skills of one programmer to the whole industry. Additionally, when choosing a language productivity and ease of coding are rarely the only consideration, which is good. While writing Python is twice as fast as writing C++, no one is seriously considering implementing a database or operating system in Python.

5 Conclusion

References

[1] *Code Time Report*. `https://www.software.com/reports/code-time-report`. [Accessed 21-Jul-2023]. 2019.

[2] Lutz Prechelt. "An empirical comparison of c, c++, java, perl, python, rexx and tcl". In: *IEEE Computer* 33.10 (2000), pp. 23–29.

[3] *TIOBE Index - TIOBE — tiobe.com*. `https://www.tiobe.com/tiobe-index/`. [Accessed 21-Jul-2023]. 2023.