An Empirical Investigation Migrating From Python 2 to Python 3

Pete Aguirre II
aguirp1@unlv.nevada.edu
University of Nevada, Las Vegas

ABSTRACT

The End of Life (EOL) for Python 2 was first announced in 2015 and the EOL for Python 2 was in January 2020, leaving Python 3 as its successor. This gives Python developers almost five years to be prepared. But how does that impact experienced Python programmers to the new ones? While the ship has sailed, switching to Python 3 has benefits, assuming a Python developer is aware of the changes and features that were added in Python 3. With 5 people as the sample population, we were able to analyze whether the switch to Python 2 to Python 3 was beneficial to the Python community. We conclude that Python 3 was more beneficial towards experienced Python programmers as they knew how to work with the new features that are in Python 3.

CCS CONCEPTS

• **Mathematics of computing** → **Statistical software**; • **Applied computing** → **Education**; • **Human-centered computing** → *Empirical studies in HCI*; • **Software and its engineering** → *Domain specific languages.*

KEYWORDS

comma, separated

ACM Reference Format:
Pete Aguirre II. 2019. An Empirical Investigation Migrating From Python 2 to Python 3 . In ,. ACM, New York, NY, USA, 4 pages. https://doi.org/555

1 INTRODUCTION

Many programming languages evolve overtime, causing other communities to have to adjust their coding style. These include changes based on external and internal factors (i.e. syntax trends, security flaws, new hardware). However, changes can either be positive or negative with users. Urma explains, "evolving programming languages is however challenging at various levels. Firstly, the impact can be negative"[5]. Let's take Python's case as an example, where the two programming languages of Python 2 and Python 3 are incompatible with one another. If these two language versions are incompatible, developers must choose to either co-evolve their code base or reject the new language version. Urma describes in one article, co-evolving code bases to be expensive and rejecting a new language could leave implications.[5] In this case, Python, in 2015, announced the End of Life (EOL) for Python 2.7 to be set on January 1st, 2020. This will force Python developers to do a full switch to Python 3. Although five years was given to prepare for a full transition, it remains unclear whether the switch to Python 3 affects developers and students positively or negatively.
The goal in this study is to discover if students and developers are affected positively and negatively by the transition from Python 2 to Python 3. We will take away three important differences that makes Python 2 and Python 3 different. Then analyze which changes are challenging to the participants.

2 BACKGROUND AND RELATED WORK

The first Python language was created in the late 1980s by Guido van Rossum. The language's philosophy is summarized in the document The Zen of Python (PEP 20), which addresses subjective claims, like:

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

Guido van Rossum envisioned Python's design to be highly extensible. Meaning that in the future, addition of new functionality and modification of existing functionality will take little effort. Guido van Rossum then released Python 2 in October 2000, introducing its own garbage collection system. Python 2 continues to trend in 2003 to the present day. Python 3 followed on December 3, 2008. On January 1, 2020, Python 2's End of Life (EOL) has been announced and will no longer be supported. It is encouraged that developers move to Python 3 as soon as possible [1].
The differences between Python 2 and Python 3 has not been closely studied. However, there are studies that quantify the transition between Python 2 to Python 3. In Malloy and Power's study, they analysed different Python applications using PyComply to measure and quantify how much Python 3 features were being used. They found that Python developers has not made a full transition from Python 2 to Python 3. [4]
Regarding of errors, Tobias Kohn did a research study on the cause of errors and the interactions between users and the compiler using the programming language Python. [3]
There is also a study by Raoul-Gabriel Urma in regards of programming language evolution. In this study, Urma mentioned the changes between Python 2 and Python 3. [5]

3 METHODS

This section will describe the methods that will be used for this study following CONSORT style to report trials.

No.	Task
1	Data Types
2	Division Operator
3	Byte vs Unicode

Table 1: List of Task

3.1 Trial Design

This experiment is conducted using repeated measures controlled trial with four tasks using the programming language Python. Depending on what group participants are assigned to, they will either use Python 2 or Python 3 for their task. What we plan to gather from each participant is: whether if a given task is more difficult to implement in Python 2 rather than Python 3 or vice versa. For this purpose, we will reach a conclusion to our research question regarding the importance of the changes made in Python 2 to Python 3. A hopeful scenario for this experiment is to gather students who are starting to learn Python or has an experience with at least one programming language.

3.2 Participants

A requirement for this experiment is for the participant to have an experience with at least one programming language. Ideal candidates for this experiment would be undergraduate computer science students. Because of current restrictions, participants will be recruited via online.

3.3 Interventions

Participants were proctored via Repl.it, an online IDE. One of Repl.it's feature is collaborative coding, which makes it easy to track: what they are typing, compiler errors, and if they have completed the task without using screen share. Participants are provided with the following: instructions regarding about the tasks, four tasks (including a practice run) and a cheat sheet (for either Python 2 or Python 3). Participants are then given fifteen minutes to complete each task.

3.3.1 Tasks. For this experiment, I decided to focus on three main changes that were updated from Python 2 to Python 3. Next, we also investigated the most common compiler errors in Python, which were gathered in one of Tobias Kohn's works [3] [2]. These task were tested to verify if the experiment will be sufficient to gather data for our research.
Since resources are limited during this time, participants were handpicked. However, they are still undergraduate computer science students.
During the course of review, the task that were decided are in Figure 1 below. These task were handpicked based on the biggest changes from Python 2 to Python 3 and have gone through feedback from the class.

3.4 Outcomes

First, participants will be timed on how much time they spend on each task and the compiler errors they commit will be counted. With this information, we can see if a Python version is more challenging than the other.
Next, the groups using Python 3 will have their code reviewed whether if they used the features that were implemented for Python 3. However, Task 1 will be ignored since there is only one way to do a print statement in Python 3. Tasks 2 and 3 will be analyzed and conclude whether if Python 3 features were utilized or not.

3.5 Sample Size

For this study, we aim to have a sample size of 14 participants, 7 participants in Group 1 and 7 participants in Group 2.

3.6 Randomization

Participants will be assigned to group 1 and 2. Group 1 using Python 2 and Group 2 using Python 3.
Groups are assigned by alternating depending on the previous participant. The first participant is automatically assigned to Group 1, then the next participant will be assigned to Group 2, and so on.

3.7 Blinding

Cheat sheets for Python 2 and Python 3 can be easily obtained online. These two differences are normally shown together side by side. Since this could effect the blinding of the experiment, it's best to separate these two styles manually. There are different ways to do each tasks that the proctor might not know. This creates a double blind study.

3.8 Statistical Methods

For statistical analysis, we performed the following tests: T.Test, Covariance, and repeated measures ANOVA. Comparing group vs time in seconds, language vs time in seconds, group vs errors, and group vs errors.

4 RESULTS

4.1 Participant Flow, Losses, and Exclusions

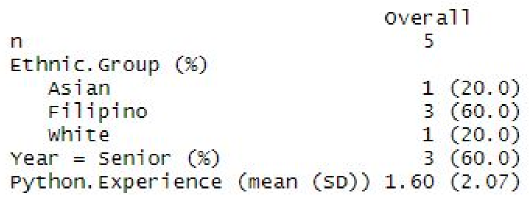
Initially, 10 people initially were recruited to volunteer for the study. However, 5 people were able to participate and complete the experiment. All participants all majored in Computer Science. The 5 participants were randomized and assigned a group. Three participants were assigned in group 1, which they had to do the first three task in Python 2, then the last three task in Python 3. Two participants were assigned in group 2, first three task in Python 3, then last three task in Python 2.

4.2 Recruitment

Web based recruitment started approximately April 2020.

4.3 Baseline Data

Table 2 shows the demographics of this study. The data distributes across ethnicity, academic year, and python programming experience (in years).



```
                                     Overall
n                                       5
Ethnic.Group (%)
   Asian                                1 (20.0)
   Filipino                             3 (60.0)
   White                                1 (20.0)
Year = Senior (%)                       3 (60.0)
Python.Experience (mean (SD)) 1.60 (2.07)
```

Table 2: Baseline Characteristics

4.4 Numbers Analyzed

4.4.1 Time. Overall, we analyzed 30 different task times that were measured in seconds. As mentioned, 3 participants were assigned to Group 1 and 2 participants were assigned to Group 2. These two groups had different conditions. Group 1, those who did Python 2 tasks first, had the lowest mean time in seconds (M=292.22, SD=262.93). Group 2, those who did the Python 3 tasks first, had the highest mean time in seconds (M=293.5, SD=229.94). Now, comparing the two languages, Python 3 the highest mean time in seconds (M=328.07, SD=279.71). In contrast, Python 2 had the lowest mean time in seconds (M=257.4, SD=221.12).

4.4.2 Quantitative Error Analysis. Quantitative error analysis is described in subsection *3.3.1*. From 30 events, compiler errors were counted in the observations. Group 1 had the lowest mean errors count (M=1.11, SD=2.03). Group 2 had the highest mean errors count, but the difference is not too far (M=1,92, SD=2.39). Comparing the two languages, in errors, Python 3 had the most mean error count (M=2.13, SD=2.77). Python 2 had the least mean error count (M=0.73, SD=1.03). Most of the errors are in Python 3, found in Task 6 for Group 1 and Task 3 for Group 2, as shown in Figure 2. Additionally, inexperienced Python programmers found it difficult to complete the third task for Python 3 for both groups.

4.5 Outcomes and Estimation

In this section, we will present the analysis of the experiment outcomes of: time and quantitative errors.

4.5.1 Task Completion by Time. Figure 1 are two boxplots separating Group 1 and Group 2 by task number and time of completion for the given task given in seconds. A repeated measures ANOVA was used to compare time task completion by language used. Data suggests that there is no significant difference between time completion by language $F(1,4) = 1.472$, $p = 0.292$, $\eta_p^2 = 0.065$ at significance level $\alpha = 0.05$. The difference between Python experience is also not significant $F(1, 3) = 8.088$, $p = 0.065$, $\eta_p^2 = 0.729$ at significance level $\alpha = 0.05$. Mauchly's Test for Sphericity was not needed because within-Ss variables were not >2 levels.

4.5.2 Quantitative Errors. To further solidify our analysis, we ran a repeated measures ANOVA for number of errors committed by language and Python experience. Results suggest that the difference between language is significant $F(1, 4) = 9.586957$, $p = 0.036$, $\eta_p^2 = 0.29$ at significance level $\alpha = 0.05$. Another sets of results also suggest that the difference between Python experience is significant $F(1, 3) = 10.195$, $p = 0.0496$, $\eta_p^2 = 0.773$ at significance level $\alpha = 0.05$.

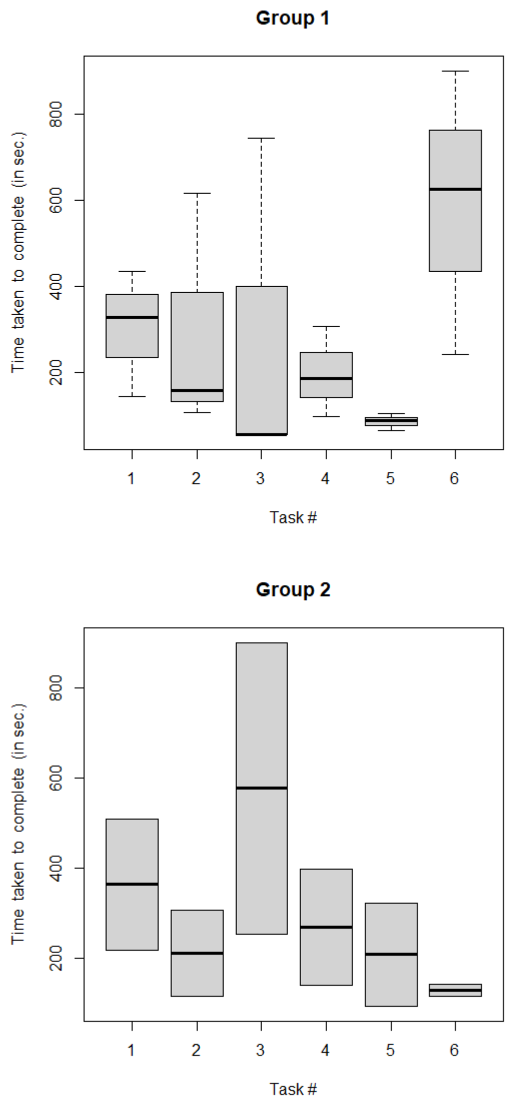


Figure 1: Time taken to complete task

5 DISCUSSION

5.1 Limitations

There are several important limitations that needs to be mentioned in the study. The small sample size and the participants lack of experience between the two languages is worth mentioning. Outcomes of measuring productivity level might be more clear and decisive with experienced Python developers.

All test were given in a manually, meaning the experiments had to be proctored in able to keep track of: time and errors. This made some of the participants feel uneasy. This also limited the study to have limited participants since they were not able to freely do the experiment in their own time.

Even if we do find an answer to the study, there is some ambiguity

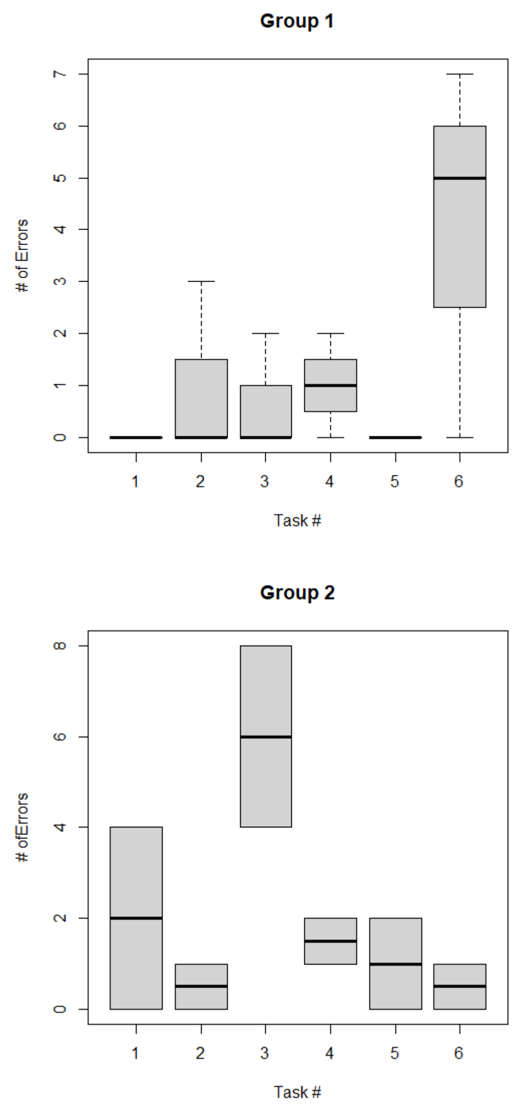


Figure 2: Errors committed by task

with the Python language. For instance, Python 3 features were not fully utilized.

5.2 Generalizability

As mentioned before, having a proctor made some participants nervous. Participants may have been trying to make less mistakes, which gave us our current results. However, the more experienced Python programmers did not report any nervousness. This might have played a factor with the results.
Inexperienced Python programmers were also not aware of the different versions between Python 2 and Python 3.

5.3 Interpretation

The main question of this study is: were the changes of Python 2 to Python 3 worth it? Looking at the two means of time completion and errors committed, Python 2 is the clear winner. However, there is no evidence to support language affect time completion of a task. On the other hand, there is evidence that support language affecting errors committed. In this case, Python 3 tasks has more errors produced than Python 2 tasks.
As for the migration between Python 2 to Python 3, initially, all but one participant preferred Python 3 over Python 2. One participant that preferred Python 2 over Python 3 claimed that Python 3 had more work for some task in contrast of its Python 2 counterpart. However, after showing the participant some Python 3 features, that participant sees why Python 3 is beneficial.

6 CONCLUSION

This paper presented a study regarding an empirical study migrating from Python 2 to Python 3 with 5 participants, by comparing Python 2 and Python 3. With our analysis, we found no significance between the language, time completion, and Python experience. Same results with groups, time completion, and Python experience. However, there was a significant difference between the number of errors, language, groups, and Python experience. With this information, we can say that the productivity level of a Python programmer is based on experience. The more experience you have, the less mistakes a programmer is bound to make.

6.1 Protocol

The code and replication packet for this study is submitted to Andreas Stefik, Ph.D.

REFERENCES

[1] Editor. [n. d.]. Python 2.7 To Be Maintained Until 2020. *i* ([n. d.]). http://www.i-programmer.info/news/216-python/7179-python-27-to-be-maintained-until-2020.html

[2] Tobias Kohn and Bill Manaris. 2020. Tell Me What's Wrong: A Python IDE with Error Messages. (2020), 1054–1060. https://doi.org/10.1145/3328778.3366920

[3] Tobias Kohn, Tobias Kohn University of Cambridge, University of Cambridge, Union County College, University of North Carolina, North Carolina State University, and Texas Woman's University. 2019. The Error Behind The Message: Finding the Cause of Error Messages in Python. *The Error Behind The Message | Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Feb 2019). https://dl.acm.org/doi/10.1145/3287324.3287381

[4] B. A. Malloy and J. F. Power. 2017. Quantifying the Transition from Python 2 to 3: An Empirical Study of Python Applications. (2017), 314–323.

[5] Raoul-Gabriel Urma, Dominic Orchard, and Alan Mycroft. 2014. Programming Language Evolution Workshop Report. (2014), 1–3. https://doi.org/10.1145/2717124.2717125