Improving Python Error Messages for Novices.

Mandisa Tunzi
Department of Computer Science,
University of Cape Town,
Private Bag X3, Rondebosch, 7701,
South Africa
tnzman002@myuct.ac.za

ABSTRACT

Python is a widely used programming language that is popular amongst novices and in many introductory programming courses. However, one of the significant challenges that novice programmers face is understanding the language's error messages. As the primary source of feedback, error messages can be a barrier to learning if novice programmers struggle to interpret and respond to them effectively. This review will examine the existing research on Python's error messages and their impact on novice programmers' learning.

The review begins by discussing the importance of error messages in programming. It discusses the common errors made by novice programmers and the challenges they pose to them. It then discusses some of the factors that affect the effectiveness of the error messages, such as readability. The review then explores the strategies and techniques proposed by researchers and practitioners to improve the effectiveness of Python's error messages, such as enhancing error messages through automated tools, and explores the possibility of presenting these errors in other languages spoken by non–native English speakers. Furthermore, the review examines the empirical studies that have investigated the impact of Python's error messages on students' performance. Finally, the review synthesizes the findings from the existing literature, identifies gaps, and proposes directions for future research. It concludes Python's error messages present a significant challenge to novice programmers' learning, and more research is needed to enhance their effectiveness, and minimize their negative impact on student's performance.

KEYWORDS

[Interpreter; error messages; enhanced error messages; common errors; novice; Python; novice programmers; programming in native languages]

1 INTRODUCTION AND MOTIVATION

Learning how to program can be a challenging task. An important aspect of this learning process is making mistakes and correcting them. Interpreter error messages facilitate this aspect by providing programmers with information on where and what went wrong in their code. However, interpreters are known to be cryptic and difficult to understand, causing confusion and frustration among programmers [8]. The work done by Prather et al. [2017] and Barik et al. [2017] provides empirical evidence that programmers spend a significant amount of time trying to understand error messages and that this impacts their overall task performance. This highlights the importance of error messages for programmers and the need to improve them. Motivated by the desire to alleviate the confusion and frustration to novice programmers, there has been an increase in research to determine ways to improve error messages. This paper assesses the approaches that have been explored particularly in the context of Python error messages in the last 5 years. This includes the effects of enhancing standard Python error messages [3, 7, 8]; factors that contribute to the readability of error messages [4, 9, 10] and the possibility of newer techniques, such as localizing error messages [11, 13] to improve error messages. The purpose of this investigation is to ultimately determine how to present Python's default error messages in a clear, and possibly multilingual way, to improve understandability to programmers, drawing from the literature the different methods and approaches that have been explored. As a widely used language, improving the understandability of error messages in Python can improve the learning experience for novice programmers, and ultimately contribute to the development of more effective teaching methods for programming.

The effectiveness of enhancing standard Python error messages has been inconclusive, with some studies showing positive impact and others showing no improvement. One possible reason for these mixed results is the lack of a standardized approach to designing error message enhancement. Researchers use different methods to enhance error messages for their respective studies, making it difficult to compare and draw generalized conclusions from the results [12].

Readability is defined as how easy it is for someone to read and understand a text [10]. Although the literature does not say how to achieve or assess it, it has been established as an essential characteristic to improve error message understandability and usefulness. This is also evident in the frequency in which it is quoted by researchers in the literature – most of them providing factors that affect readability.

Using the rationale that programming is a skill that is learned by many non-English speaking individuals around the world, this paper considers the impact of localizing error messages, which refers to the effect of presenting error messages in a language spoken by the programmers, instead of purely in English as done currently. Although localizing error messages of programming languages has been the topic of some research papers [1, 11, 13], there has been no work done to investigate the impact of doing this for interpreter error messages.

Focusing on the Python programming language, this paper aims to determine a more specific and target approach to error message improvement, drawing from the results discussed in these sections.

This paper is laid out as follows: In Section 2 the methods and research questions used for reviewing the literature are stated. A review of the literature is conducted, and the results are presented in Section 3 under the different themes identified in the reviewed work. The results in Section 3 are discussed under the 4 research questions. Section 5 concludes the paper.

2 METHODS

This section describes the research questions, the criteria and the search process used to ultimately complete this review.

2.1 Research questions

The following four research questions guide this review:

RQ1:
What is the challenge faced by novice programmers and what is the impact of error messages on novices' programming performance?

RQ2:
What factors contribute to the understandability of error messages; how can developers improve error message design?

RQ3:
What approaches have been proposed for presenting error messages in a possibly multilingual way?

RQ4:
What is the empirical evidence for the effectiveness of tools designed to improve the understandability of error messages?

2.2 Search Process

The search for the papers presented in this paper was done in two phases. The first phase began by reading the work of Becker et al. [2019], which synthesized recent information gathered on the topic thus far. I then gathered all the relevant papers (using keywords above as a guide) they cited in their paper and later the references of the discovered papers. I then proceeded to read title and abstracts to verify relevance – resulting in some papers being discarded. For the second phase, I used search engines: google scholar, ACM digital library, IEEEXplore and arxiv to search for more journal papers on the topic using the keywords stated above. After finding the papers, I started by removing all the duplicates from the different engines. I then read through their titles, abstracts and conclusions to determine their relevance as earlier – leading to more papers being discarded. The remaining papers were read fully. A few were discarded and the remaining papers, were included in this literature review, totalling 17 papers.

2.3 Criteria

The review includes journal articles and conference papers available on the topic of the effectiveness of error messages and improving them– focussing on Python: The keywords above were used on search engines to look for the papers and when reading through the reference lists of already found papers**.** The papers included were published between the period of 2017 - 2022, except for 2 which talk about common error messages and were published in 2015. All the papers that were found and included, were at least 5 pages long.

3 RESULTS

This section summarizes and presents the findings gathered from the papers included in the review which will be used answer the research questions posed in Section 2.1.

3.1 Common errors made by novices.

In an in-depth study conducted by Kohn [2019] a total of 4091 instances of Python error messages produced by programs collected from high school students taking an introductory programming course were analysed. The results of the study were organized into a frequency table which revealed that minor mistakes constitute a significant part of the collected errors. They made 30% of the total, and all were syntax errors, mostly manifesting as name and parentheses errors. Earlier work by Prichard [2015] presents the results of analysing 640 000 errors from 1.6 million code submissions from a Python introductory course. This study recorded the errors: "Syntax: Invalid tax" followed by "Name error: name "NAME" is not defined" (a type of syntax error) as the most common errors in Python. Finally, Altadmri et al [2015] present the result of a study of 17 682 007 errors of students across many institutions collected over a period of a year. The results show that parentheses error, (a type of syntax error) was the most frequent error followed by a type error, (a type of semantic error) in a frequency table. They also showed that in general, syntax errors were only prevalent in the earlier stages of programming, but as programmers advanced, semantic errors became more significant.

3.2 Impact on performance

Investigations on how novice and intermediate programmers interact with compiler error messages shows that they struggle to understand messages and often misinterpret them [8]. Through examining developer fixations, using an eye – tracking device, Barik et al. [2017] discovered that students spend a significant amount of time trying to understand error messages. The study revealed that while the programmers did indeed read the error messages, they spent a long time trying to read a specific error, and thus they took more time to fix a coding error or complete a programming task [15]. The paper also analysed revisit times (going over the same error) as a proxy for the difficulty of reading the error messages. The results suggest that the difficulty of a task and the amount of time spent on it can be attributed to the difficulties that programmers experience when they try to read and understand error messages, as there were increasing incorrect tasks associated with increasing revisits [15].

The following Section presents findings on readability, which has been identified as an important factor that contributes to understanding error messages.

3.3 Readability

Several authors in the literature have emphasized the importance of error message readability, but its impact on improving error messages remains unclear [5], with the literature lacking in concrete criteria and empirical evidence for achieving and assessing it. Existing work [4, 9, 10] emphasize this issue and the significance of error message readability in programming refer to readability as how easy it is for someone to read and understand a text [10], with the text being the error message. For instance, Danny et al. [2020] provides evidence for the perceived usefulness of enhanced error messages that adhere to the following guidelines:

- "Reduce cognitive load",
- "Show solutions for hints",
- "Use a positive tone", and
- "Increase readability".

The result of the study revealed that programmers took less time to solve bugs when they used the enhanced error messages designed according to these guidelines. Readability in the paper was analysed on two metrics namely the Flesch reading score, using syllables and sentence length to measure readability; and the Dale-chall score, using a dictionary of easily understood words to compare and measure readability in the enhanced error messages. Becker et al. [2021] and Denny et al. [2021] investigate the factors that contribute to error message readability, identifying a collective of:

- message length,
- use of syntax,
- tone,
- sentence structure,
- and vocabulary,

as key factors affecting readability. This was determined by ratings of expert, non-expert, and novice students.

The following Section presents findings on an area of research relating to approaches for presenting programming languages in an alternative way, as opposed to just English.

3.4 Localizing error messages

While there might not be much literature specifically examining the impact of localizing error messages, there is research on the broader issue of localizing programming languages that could inform our results. This includes the work of Guo [2018] who investigates the barriers faced by non-native English speakers when learning how to program; and the work of Dasgupta and Hill [2017] and Raj et al. [2018] who conduct empirical investigations on the influence of learners' native language on their ability to learn a programming language. The study by Guo [2018] analyzed 840 responses to a survey spanning 86 countries and 74 native languages and the results suggest that non – native English speakers face several barriers when learning programming including:

- reading instructional materials,
- technical communication,
- reading and writing code and
- learning English and programming at the same time.

The study by Dasgupta and Hill [2017] shows that localizing programming languages (presenting them in a language spoken by the programmer) can have a positive impact for non – native speakers. It found that Italian-speaking novice programmers performed better and showed a more positive attitude when using a localized version of Scratch, a visual programming language, compared to using the English version. Raj et al. [2018] reported Positive reactions from Tamil-speaking students who were taught programming in their native language and English, although showing no significant differences between grades obtained by those being taught purely in English and those being taught in both English and Tamil.

The following section presents findings on enhancing error messages to improve their readability.

3.5 Error message Enhancement

There is a significant body of literature on enhancing compiler error messages to improve the usability and effectiveness of compilers for software developers. For this review, I define Enhancement as the action of a tool, to improve, the error message presented by another tool [5], usually by changing, or adding to the content of the error message. There are varying results among the studies that investigate the effect of enhancing error messages. While some studies report improvements [3, 7, 8], others do not [12, 17] even though the questions used for the studies are similar.

4 Discussion

This section provides a discussion on the findings presented in Section 3. It is organized according to themes of the 4 research questions.

4.1 Challenge faced by novice programmers and the impact of error messages on novices' programming performance.

In understanding the impact of error messages on novice programmers and be able to measure the impact of improving error messages, it is important to understand the types of errors made by novices. This has been investigated by many researchers in the past with the general result suggesting that students, especially novices, make numerous syntax errors and often have a problem locating and fixing these errors using standard error messages [5]. The papers presented in Section 3.1 reveal some of the recent work in this regard. The work done by Prichard [2015] and Kohn [2019] both reveal syntax errors as the most common errors in Python, specifically missing parentheses errors, name errors, and other errors resulting in the error message, "Syntax error: invalid syntax". Prichard [2015] attributes the frequent occurrence of syntax errors to the presented message, "Syntax error: invalid syntax", being generic and appearing for different errors. Although Altadmri et al. [2015], tries to challenge the notion of syntax errors being the most important factor in programming education, stating that semantic errors are more serious errors, their study also revealed that syntax error were more frequent in the earlier stages of learning how to program. The significance of these papers is their coverage of large datasets to investigate the common errors made by novices. They inform literature of where exactly to investigate in terms of understanding novices' challenges with error messages.

It is also important to understand how error messages impact novice users before establishing ways to improve them. This will help in gaining insight into how we can improve them in an effective manner. Studies by Prather et al, [2017] and Barik et al. [2017] present some of the recent work done to investigate this impact. Although their studies use the Java programming language, this paper draws from their general conclusions about how novice and intermediate programmers use error messages. Their conclusions apply to this paper because the focus of the studies was to determine the relationship between programmers and error messages as opposed to Java itself. Their findings highlight the fact that error messages can be a significant barrier for novice and intermediate programmers, hindering their ability to understand and ultimately fix code errors. Their findings underscore the importance of improving the error messages, to improve the performance of novice/ intermediate programmers.

4.2 Factors that contribute to the understandability of error messages: Readability.

Readability is an important property of effective compiler/ interpreter error messages. This is evident in it being quoted in some way in many of the papers that discuss guidelines for effective error messages [5]. The work done by Denny et al. [2020], presented positive results in improving the readability of error messages. However, this finding cannot be generalized as the analysis of whether the newly designed error messages were indeed readable used metrics that were not designed for error messages. The paper does however provide a little objective evidence to support that error messages may be formatted in a way that is easier to read and that this may affect how well the error is understood by the programmer. In the study by Becker et al. [2021], the factors affecting readability may be considered plausible as they match those presented in other literature, but they result from a small subset of error messages in the conducted study and do not elaborate on the extent to which readability is impacted by these factors. Following this work, Denny et al.

[2021] provide this elaboration, identifying strong correlations to readability with each of the factors.

Overall, none of these papers define or create a metric to measure readability, making it difficult to generalize the information. They do however emphasize the importance of readability and lay the groundwork for further research to determine the impact of readability guidelines, which are currently difficult to measure [10].

4.3 Presenting error messages in a multilingual way.

With the large percentage of the world being non-native English speakers, there is a gap in literature to investigate the impact of having error messages entirely in English and whether changing this would improve their understandability. So far there has not been any work done in literature to investigate the impact of specifically changing the language in which error messages are presented. None of the papers presented in Section 3.4 address error messages specifically. However, we can draw from their results, as error messages form a significant part of learning how to program, which is the broader domain they cover. In the study of Guo [2018], the Scratch programming editor, website and programming blocks (the equivalent of tokens in text-based programming languages) were all translated into different Latin languages which were selected based on the user's web configuration. The translations were provided by volunteer translators using a web – based translation portal. Alternatively, Raj et al. [2018] used code-switching (switching between languages in the same conversation) and translanguaging (which refers to doing certain tasks purely in one language and others purely in the other). Code-switching helped students reinforce concepts to understand better and tranlanguaging helped with improving their English [1]. Both use translation as an approach to provide a multilingual interface. Collectively, these studies try to determine the extent to which native languages affect novice programmers and whether translating programming languages will improve their programming learning experience. Although all report promising results in improving program understandability due to localization of the programming languages, the results presented by Dasgupta et al. [2017] are based on self-report data. This is the same for Raj et al. [2018] whose positive results are from open-ended feedback as their technical tests showed no significant difference between the two study groups. It is important to conduct more empirical studies to evaluate the impact of using native languages in programming that can make general conclusions, free from the subjectivity of self-report data.

4.4 Empirical evidence for the effectiveness of tools designed to improve the understandability of error messages.

The empirical evidence for the effectiveness of tools in the studies to improve the understandability of error messages is based on the experimental evaluations conducted in each study. Studies [3, 7, 8] evaluated the effectiveness of their proposed tools by measuring the participants' understanding of the error messages and their ability to fix syntax errors in Python code. The results consistently showed that the tools designed to improve the understandability of error messages led to improvements in the participants' understanding of the error messages and reduced the time and effort required to fix errors. Specifically, the studies evaluated methods such as utilizing external sources like stack overflow to add and customize error messages [7], providing specific information about the location and cause of errors [8] and designing error messages that provide clear and concise information [3]. The empirical evidence presented in these studies suggests that these tools can significantly improve the understandability of error messages, leading to more efficient and effective debugging processes.

The study Zhou et al. [2021] also evaluates the effectiveness of an automated tool used to enhance Python error messages by measuring if the error messages help students reduce the number of errors they make and whether their debugging perf romance improves. However, this study produced opposing effects to the 3 above, finding that enhanced programming error messages did not help to reduce students' errors or help them improve their debugging. The empirical significance of this study lies in the large dataset, having analyzed 6339 student solutions to deduce the findings. Pettit [2017] also came to this conclusion in his study, however measuring the likelihood of successive compilation errors. Its empirical significance also lies in its large dataset gathered over a period of 4 semesters. Larger data sets allow for more generalizable results.

Interestingly, in all these studies (with positive and opposing results), most only noted that they used some automated tool to enhance the messages, but never defined or discussed the specific ways in which they enhanced their error messages. Only Thiselton et al. [2019], presented in this section does this by explaining that they enhance the error messages by customizing and simplifying results from stack overflow. Other work that are not included in this section include:

Paul [2021] who enhance the error messages through shorter text and giving hints on how to solve,

Barik [2018] who enhance the error messages through LLM.

The significance of this is that the experienced differences in the findings above may be a consequence of the differences in what exactly is enhanced in the error messages in the different studies. Two studies showing different results while using the same questions to evaluate may be an indicator of the differences in the tools used. Maybe some ways of enhancing error messages are more effective than others, but this cannot be determined unless they are explicitly defined/ described and the impact of each evaluated.

5 Conclusions

Using RQ1, we conclude that the impact of error messages on novice programmers necessitates the need for improving them to enhance the programmers' learning experience. The results presented suggest that a good place to improve python error messages, such that they have meaningful impact is by improving their syntax error messages, as the identified common problem amongst novices.

Under Q2, we conclude that improving readability can lead to better understanding and performance by novice programmers. However, further research is needed to develop metrics for measuring readability and to determine the impact of implementing readability guidelines in error messages. Overall, readability and its guidelines should be considered when designing error messages or tools to improve them for novices.

For Q3, based on the information provided, it can be concluded that there is a gap in the literature regarding the impact of

presenting error messages in a multilingual way. Further empirical studies are needed to draw more generalized conclusions regarding the impact of localizing programming languages in learning how to program. Overall, the studies suggest that providing multilingual interfaces in programming languages may improve novice programmers' learning experience, but more research is needed to its effectiveness and how it can be incorporated to improve error messages.

Finally, under Q4, the empirical evidence presented in the studies in Section 4.4 suggest that tools designed for enhancing error messages can significantly improve the efficiency and effectiveness of understanding and fixing errors. However, these tools can also reproduce results suggesting that they don't make a difference. This difference in findings may be due to the specific ways in which the error messages are enhanced, which are not always explicitly defined or discussed in the studies. Therefore, it is important to define and evaluate the impact of each method of enhancing error messages to determine which methods are more effective.

6 REFERENCES

[1] Adalbert Gerald Soosai Raj, Kasama Ketsuriyonk, Jignesh M. Patel, and Richard Halverson. 2018. Does native language play a role in learning a programming language? *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (Feb. 2018), 417–422. DOI: https://doi.org/10.1145/3159450.3159531

[2] Amjad Altadmri and Neil C.C. Brown. 2015. 37 Million Compilations: Investigating Novice Programming Mistakes in Large-Scale Student Data. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (Feb. 2015), 522-527. DOI: https://doi.org/10.1145/2676723.2677258

[3] Brett A. Becker, Kyle Goslin, and Graham Glanville. 2018. The Effects of Enhanced Compiler Error Messages on a Syntax Error Debugging Test. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (Feb. 2018), 640–645. DOI: http://dx.doi.org/10.1145/3159450.3159461

[4] Brett A. Becker, Paul Denny, James Prather, Raymond Pettit, Robert Nix, and Catherine Mooney. 2021. Towards Assessing the Readability of Programming Error Messages. *Proceedings of the 23rd Australasian Computing Education Conference* (March. 2021), 181-188. DOI: https://doi.org/10.1145/3441636.3442320

[5] Brett A. Becker, Paul Denny, Raymond Pettit, Durell Bouchard, Dennis J. Bouvier, Brian Harrington, Amir Kamil, Amey Karkare, Chris McDonald, Peter-Michael Osera, Janice L. Pearce, and James Prather. 2019. Compiler Error Messages Considered Unhelpful: The Landscape of Text-based Programming Error Message Research. *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education* (Dec. 2019), 177-210. DOI: https://doi.org/10.1145/3344429.3372508

[6] David Pritchard. 2015. Frequency Distribution of Error Messages. *Proceedings of the 6th Workshop on Evaluation and Usability of Programming Languages and Tools* (Oct. 2015), 1–8. DOI: http://dx.doi.org/10.1145/2846680.2846681

[7] Emillie Thiselton and Christoph Treude. 2019. Enhancing Python Compiler Error Messages via Stack. *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* (Oct. 2019), 1-12. DOI: https://doi.org/10.1109/esem.2019.8870155

[8] James Prather et al. 2017. On Novices' Interaction with Compiler error messages: A Human Factors Approach. *Proceedings of the 2017 ACM Conference on International Computing Education Research* (Aug. 2017), 74–82. DOI: http://dx.doi.org/10.1145/3105726.3106169

[9] Paul Denny, James Prather, and Brett A. Becker. 2020. Error Message Readability and Novice Debugging Performance. *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (June. 2020), 480–486. DOI: http://dx.doi.org/10.1145/3341525.3387384

[10] Paul Denny, James Prather, Brett A. Becker, Catherine Mooney, John Homer, Zachary C Albrecht, and Garrett B. Powell. 2021. On Designing Programming Error Messages for Novices: Readability and its Constituent Factors. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (May. 2021), 1-15. DOI: https://doi.org/10.1145/3411764.3445696

[11] Philip J. Guo. 2018. Non-Native English Speakers Learning Computer Programming. Programming: Barriers, Desires, and Design Opportunities. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (April. 2018), 1-14. DOI: https://doi.org/10.1145/3173574.3173970

[12] Raymond S. Pettit, John Homer, and Roger Gee. 2017. Do Enhanced Compiler Error Messages Help Students? Results Inconclusive. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (March. 2017), 465–470. DOI: http://dx.doi.org/10.1145/3017680.3017768

[13] Sayamindu Dasgupta and Benjamin Mako Hill. 2017. Learning to Code in localized Programming Languages. *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale* (April. 2017), 33–39. DOI: https://doi.org/10.1145/3051457.3051464

[14] Titus Barik, Denae Ford, Emerson Murphy-Hill, and Chris Parnin. 2018. How Should Compilers Explain Problems to Developers? *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Oct. 2018), 633–643. DOI: http://dx.doi.org/10.114X5/3236024.3236040

[15] Titus Barik, Justin Smith, Kevin Lubick, Elisabeth Holmes, Jing Feng, Emerson Murphy-Hill, and Chris Parnin. 2017. Do Developers Read Compiler Error Messages? *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)* (July. 2017), 575-585. DOI: Https://doi.org/10.1109/icse.2017.59

[16] Tobias Kohn. 2019. The Error Behind the Message: Finding the Cause of Error Messages in Python. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Feb. 2019), 524-530. DOI: https://doi.org/10.1145/3287324.3287381

[17] Zihe Zhou, Shijuan Wang, and Yizhou Qian. 2021. Learning from Frrors: Exploring the Effectiveness of Enhanced Error messages in Learning to Program. *Frontiers in Psychology* 12 (Nov 2021), 5686 pages. DOI: http://dx.doi.org/10.3389/fpsyg.2021.768962