Understanding Pythonic Idioms Via Visualization

Jacquelyn Yapenare
PNGUoT, Papua New Guinea
21302387jaya@student.pnguot.ac.pg

Kazumasa Shimari
NAIST, Japan
k.shimari@is.naist.jp

Brittany Reid
NAIST, Japan
brittany.reid@naist.ac.jp

Sankwi Abuzo
PNGUoT, Papua New Guinea
sankwi.abuzo@pnguot.ac.pg

Benson Mirou
PGUoT, Papua New Guinea
benson.mirou@pnguot.ac.pg

Raula Gaikovina Kula
Osaka University, Japan
raula-k@ist.osaka-u.ac.jp

***Abstract*—Pythonic idioms are efficient coding patterns that enhance the readability and maintainability of Python code. However, they can be challenging for some developers, particularly beginners, to grasp. This study explores whether AI tools tools like Napkin.ai and ChatGPT can generate visualizations of Pythonic idioms. Pythonic code snippets are input into these tool and prompted to generate visual representations of the codes. A simple ranking method analysis is conducted to determine whether the tools successfully translated the Pythonic codes into visualization, while also identifying which AI tool generated a more effective visual output.**

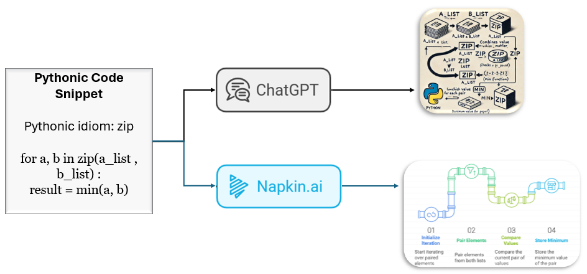
***Index Terms*—Pythonic Idioms, AI Tools**

I. Introduction and Motivation

Pythonic idioms including list comprehensions and generator expressions are coding patterns in Python that allow developers to write more efficient and concise code [1]. These idioms leverage Python's unique features to reduce the amount of code needed for common tasks, improving readability, performance, and memory management. Despite their many advantages, these patterns can sometimes be difficult for beginners or even experienced developers to grasp [2]. The compactness of Pythonic code, while making it more efficient, often sacrifices clarity, making it harder for developers to fully understand its behavior.

Previous studies have introduced tools such as Teddy [3] and RIdiom [4] to refactor non-Pythonic code into concise, efficient Pythonic patterns, improving performance and readability. These tools automatically suggest changes to make code cleaner and easier to understand. Conversely, tools like DeIdiom [2] convert Pythonic code back into non-idiomatic, more detailed forms to help beginners better understand the logic. In addition to refactoring and deconstructing, visualization tools, such as diagrams or flowcharts, offer a clear graphical way to understand Pythonic idioms, complementing other tools by making complex code more comprehensive to programmers [5], [6].

This preliminary experiment investigates whether AI-powered tools can generate visualizations from Python codes, with a focus on Pythonic idioms. The goal is to determine whether the visuals generated can capture the key logic and structure within the code. Evaluation is carried out by ranking the output of each input from the respective AI tool.



```
for a, b in zip(a_list ,
  b_list) :
result = min(a, b)
```

Figure 1: The process of translation of Pythonic idioms to visuals

This research questions was devised to guide the investigation:

- **RQ1:Can the AI-powered tools Napkin and ChatGPT capture Pythonic idioms and generate efficient visualizations?**

II. Methodology

A. Data Collection

Pythonic idiom code snippets from an existing paper [7] were fed as input into the AI tools ChatGPT and Napkin.ai to generate visualizations. See Figure 1. For ChatGPT, it could not generate an image without being prompted to, so a command prompt to assist was created and used. Napkin.ai allowed direct copy-pasting of code snippets without the need for a prompt to generate. It naturally produces a range of diagrams each time, with the visual layout remaining the same but the data varying depending on the input. Each code snippet was pasted and run on the different tools, thus resulting in 18 visualizations, 9 from ChatGPT and 9 from Napkin.ai. https://shorturl.at/AqmQT

"I want you to act as a software engineer. I will provide a Python code snippet, and it will be your job to generate an image or flowchart of this code, explaining the function, variables and key concepts behind the code. Code: <Code Snippet>"

–ChatGPT Prompt

Table I: Visualization Assessment Table

Pythonic Idiom	# ChatGPT	# Napkin.ai
List comprehension	2	4
Dict comprehension	2	4
Generator expression	2	3
yield	2	3
lambda	2	5
collections.defaultdict	1	3
collections.dequeue	1	4
zip	2	4
itertools	1	5

B. Analysis: Assessment of Visualizations

The evaluation of the generated visualizations was centered around determining their overall understandability. Each of the 18 visualizations from https://shorturl.at/AqmQT was rated on a scale of 1-5,where:

- 1 = Do not understand at all
- 2 = Barely understand
- 3 = Somewhat understand
- 4 = Mostly understand
- 5 = Completely understand

The goal was to assess how well each image conveyed the logic and structure of the Pythonic idioms. One of the authors, with limited familiarity with Pythonic idioms used this ranking system to evaluate all images, ensuring the result reflects the potential for these tools to assist novice programmers in understanding compact and abstract Python code. See Table I.

Observation: The results indicate that visualizations generated by Napkin.ai are more predictable.

III. Conclusion

This study explored the effectiveness of AI-powered visualization tools, specifically Napkin.ai and ChatGPT, in generating visual representations of Pythonic idioms. The findings show that both tools are able to translate Pythonic code into visual diagrams, however Napkin.ai's outputs are generally clearer and more intuitive, making it easier for novice programmers to follow understand the code. It is important to note that the Pythonic code used in this study were not real-world examples, but rather simplified snippets aimed at illustrating specific idioms. Nonetheless, the research highlights the potential of AI-generated visualizations as a helpful tool for improving comprehension of Pythonic idioms, particularly for beginners who may struggle with the compact and efficient syntax.

For future research, we aim to conduct a full study involving students, professors, educators and developers to understand the effects of these visualizations on comprehension. This will providing deeper insights into the effectiveness of these tools in real-world programming contexts. Moreover, expanding the scope to generate a larger set of visualizations, experimenting with different command prompts to generate an output and assessing their accuracy in capturing the code syntax, rather than just focusing on their understandability, would offer a more comprehensive evaluation of the tools' effectiveness.

References

[1] Z. Zhang, Z. Xing, X. Xia, X. Xu, and L. Zhu, "Making python code idiomatic by automatic refactoring non-idiomatic python code with pythonic idioms," in *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2022, p. 696–708.

[2] Z. Zhang, Z. Xing, D. Zhao, Q. Lu, X. Xu, and L. Zhu, "Hard to read and understand pythonic idioms? deldiom and explain them in non-idiomatic equivalent code," in *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)*, 2024, pp. 2808–2819.

[3] P. Phan-udom, N. Wattanakul, T. Sakulniwat, C. Ragkhitwetsagul, T. Sunetnanta, M. Choetkiertikul, and R. G. Kula, "Teddy: Automatic recommendation of pythonic idiom usage for pull-based software projects," in *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 2020, pp. 806–809.

[4] "RIdiom - IntelliJ IDEs Plugin — Marketplace — plugins.jetbrains.com," https://plugins.jetbrains.com/plugin/20107-ridiom, 2022.

[5] K. Shatri and K. Buza, "The use of visualization in teaching and learning process for developing critical thinking of students," *European Journal of Social Sciences Education and Research*, vol. 9, p. 71, 01 2017.

[6] "The Five Benefits of Data Visualization — Deloitte Netherlands — deloitte.com," https://www.deloitte.com/nl/en/services/tax/perspectives/bps-the-five-benefits-of-data-visualization.html, 2022.

[7] P. Leelaprute, B. Chinthanet, S. Wattanakriengkrai, R. G. Kula, P. Jaisri, and T. Ishio, "Does coding in pythonic zen peak performance? preliminary experiments of nine pythonic idioms at scale," in *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*, 2022, p. 575–579.