## Part 1:

How does a user's understanding of memory management differ between C and Python?

## Part 2:

I conducted a survey with the help of a Google form. The survey received 5 responses. The participants had to analyze memory usage in short C and Python code snippets and reflect on their understanding through Likert scale ratings.

Background:

All participants reported **1-2 years** of programming experience.

All were familiar with C and Python.

Objective questions:

| Question | Correct Answer | % Correct |
|---|---|---|
| C: Memory allocated for `arr` | 20 bytes | 100% (5/5) |
| C: When is memory released | Explicitly (free) | 0% (0/5) |
| Python: Memory allocated for `arr` | ~80-100 bytes | 60% (3/5) |
| Python: When is memory released | Automatically | 100% (5/5) |

| Statement | Mean | Median |
|---|---|---|
| "Understanding memory usage in **C** felt straightforward." | 4.6 | 5 |
| "Understanding memory usage in **Python** felt straightforward." | 2.6 | 2 |
| "I felt confident predicting memory usage in **Python**." | 2.6 | 2 |

## Part 3:

Thematic Analysis Report

1. Familiarizing Yourself with the Data

The dataset consisted of Likert scale responses from three participants—Saurabh, Tanvi, and Vansh—who evaluated their subjective experience of understanding memory management in C and Python. They reflected qualitative insights into how programmers perceive and reason about memory handling in different languages.

2. Generating Initial Codes

Each participant responded to statements about their confidence and understanding in both C and Python. Based on their responses, I assigned the following codes to capture core ideas:

- Participants gave high ratings to their understanding of memory in C, even when they answered technical questions incorrectly. This implies they felt “in control” due to the explicit nature of the code.
- Lower ratings for Python suggested that even when accurate—can produce unease or confusion for learners.
- Despite missing key concepts like `free()`, participants still rated C as more straightforward.
- Python’s automatic garbage collection seemed to create a sense of uncertainty, possibly due to its invisible processes.
- The unanimous incorrect answers about memory release in C indicate a conceptual misunderstanding about how manual memory deallocation works.

These codes emerged directly from comparing objective correctness with subjective confidence, revealing mismatches.

3. Searching for Themes

i. Illusion of Mastery in C
Participants overestimated their understanding of C due to the language's visible and explicit syntax, which may have given a false sense of clarity and control.

ii. Abstraction Brings Doubt
Despite correct answers regarding Python's memory model, participants reported lower confidence and understanding, indicating discomfort with systems that manage complexity on the user's behalf.

iii. Transparency Bias
Participants favored languages where they could see and manually manage memory, even when such visibility led to incorrect assumptions. This suggests a cognitive bias: what is visible feels more understandable, regardless of accuracy.

4. Reviewing Themes

These themes were refined for clarity and overlap. While Illusion of Mastery and Transparency Bias are closely connected, I separated them to distinguish between the emotional response (confidence) and the cognitive preference (visibility). I also verified that these themes interacted coherently: for example, participants' mistrust in Python (Theme 2) directly relates to their preference for visible control (Theme 3), while their C confidence despite errors is a direct product of both.

5. Defining Themes

- Illusion of Mastery in C
Participants reported high levels of understanding in C, despite failing to correctly identify the role of `free()`. This theme reflects a broader pattern in which explicit control and low-level syntax give a misleading sense of mastery. It may reflect overconfidence in learners who equate visibility with comprehension.

- Abstraction Brings Doubt
Python's garbage collection was well understood technically (as shown in multiple correct answers), but still received lower ratings in perceived

clarity and confidence. This suggests that learners may associate abstraction with unpredictability, even when the abstraction performs accurately.

- Transparency Bias
  This theme captures the cognitive preference for control. Participants appear more trusting of languages that expose memory management to the programmer. They seemed to value control and visibility over correctness

6. Writing the Report

This analysis sheds light on how beginner programmers perceive memory management in C versus Python. Despite Python offering simpler and more error-resistant memory behavior, participants expressed greater comfort with C- largely due to its visible, hands-on style of memory control. However, this control was misunderstood: none of the participants correctly identified the need for explicit memory release using `free()`, despite rating their understanding of C highly.

On the opposite side, Python's automated memory management- although correctly understood by most participants- was rated as less straightforward and inspired less confidence. This mismatch between perceived understanding and actual correctness illustrates how cognitive factors can influence learner's confidence.

The results suggest that programming language design and understanding should not only consider what learners can do, but also how confident and clear learners feel while doing it. Beginners may benefit from generalization in terms of reducing errors.

Research Question:
How does a user's understanding of memory management differ between C and Python?

Answer:
Participants understanding of memory management differs significantly between C and Python—not just in accuracy, but in perception as well. Participants in my user study displayed greater confidence and perceived

clarity when working with C, even though they answered some of the critical questions incorrectly. In contrast, Python's automatic memory management was more accurately understood by the participants. This highlighted a gap; participants tend to associate C's explicit memory control with stronger understanding, even when their answers are incorrect. Meanwhile, Python's automated memory management—though more accurate and less prone to error—was met with uncertainty. Therefore, understanding is influenced not only by correctness but also by how visible and "hands-on" memory processes feel to the user.