Here are some topics and their order for a non-coder's quick Python tutorial, focusing on practicality and engagement:

**1. Introduction (5 minutes):**

* **What is Python and why is it popular?** Briefly explain its uses and ease of learning.
* **Setting up Python:** Guide them through installing and running Python (choose a beginner-friendly platform like online editors or Anaconda).
* **Basic code structure:** Show a simple print statement and explain lines, commands, and indentation.

**2. Variables and Data Types (10 minutes):**

* **Storing information:** Explain variables and how they hold data like numbers or text.
* **Data types:** Introduce integers, floats, strings, and booleans with examples.
* **User input:** Demonstrate how to get input from the user using the input() function.

**3. Operations and Math (10 minutes):**

* **Basic arithmetic operators:** Show addition, subtraction, multiplication, and division with examples.
* **Comparison operators:** Introduce ==, !=, <, >, etc., for comparisons.
* **Simple calculations and formulas:** Use operators to create basic calculations like area or distance.

**4. Control Flow (15 minutes):**

* **Making decisions with** if **statements:** Explain how to control code based on conditions.
* **Branching with** elif **and** else**:** Show how to handle multiple conditions.
* **Loops with** for **and** while**:** Demonstrate repeating code for lists or specific times.

**5. Putting it all together (15 minutes):**

* **Building a mini-project:** Combine the learned concepts into a small program like a guessing game or a quiz.
* **Debugging:** Briefly explain how to find and fix errors in their code.
* **Next steps and resources:** Point them towards further learning materials and communities.

**Bonus:**

* **Show real-world applications of Python:** Briefly mention how Python is used in data science, web development, or automation to spark interest.
* **Use visuals and humor:** Incorporate diagrams, screenshots, or jokes to keep the learning process engaging.
* **Encourage experimentation:** Let them play around with the code and modify examples to solidify understanding.

Remember, this is a quick guide. Adjust the time and depth based on your audience's pace and interests. Keep it fun, practical, and focused on building confidence in their first steps with Python!