## **🧠 1. Introduction**

When you write Python programs, you often need to get input from users. There are **two main ways** to do this:

* input() – interactive input via keyboard
* sys.argv – command-line arguments

## **🧮 2. What is input()?**

* input() is a built-in function in Python used to take **interactive user input**.
* It **pauses execution** until the user types something and hits Enter.

### **✅ Example:**

name = input("Enter your name: ")

print(f"Hello, {name}!")

### **🔍 Behavior:**

* Used when the program **asks the user during runtime**.
* Returns the input as a **string**.

## **🧳 3. What is sys.argv?**

* sys.argv is a list in the sys module.
* It holds **command-line arguments** passed when you run the script.
* sys.argv[0] is the script name.
* sys.argv[1:] are the arguments.

### **✅ Example:**

import sys

name = sys.argv[1]

print(f"Hello, {name}!")

### **🔍 Usage:**

python script.py Gowtham

# Output: Hello, Gowtham!

## **🆚 4. Difference Between input() and sys.argv**

| **Feature** | **input()** | **sys.argv** |
| --- | --- | --- |
| Input method | Interactive during program execution | Passed at the time of script execution |
| Suitable for | Small scripts, learning, CLI interactivity | Automation, production, scripts with args |
| Returns | Always a string | List of strings |
| Use in production | ❌ Avoid (hard to automate) | ✅ Preferred for CLI tools |
| Requires import | ❌ No | ✅ Yes (import sys) |
| Error-prone on input | Yes (if user types invalid values) | Yes (if args are missing; need validation) |

## **🚨 5. Why Avoid input() in Production**

* Blocks automation pipelines.
* Difficult to test automatically (needs user input).
* Not suitable for cron jobs, Airflow, shell scripts, etc.

✅ **sys.argv allows passing data dynamically** and can be handled via argparse or click for even better control.

## **🧪 6. Example: Add Two Numbers (Both Methods)**

### **🔸 Using input()**

a = int(input("Enter first number: "))

b = int(input("Enter second number: "))

print("Sum:", a + b)

### **🔹 Using sys.argv**

import sys

a = int(sys.argv[1])

b = int(sys.argv[2])

print("Sum:", a + b)

**Run it like:**

python add.py 5 10

# Output: Sum: 15

## **🛠️ 7. Best Practice: Use argparse for CLI Tools**

import argparse

parser = argparse.ArgumentParser()

parser.add\_argument("name", help="Your name")

args = parser.parse\_args()

print(f"Hello, {args.name}")

Run:

python script.py Gowtham

# Output: Hello, Gowtham

## **🧾 Final Working Code: email\_generator.py**

import sys

# Check if enough arguments are passed

if len(sys.argv) == 1:

print("Usage: python email\_generator.py 'Full Name'")

sys.exit()

# Combine all words after script name into one string

full\_name = " ".join(sys.argv[1:])

# Format the name

email = full\_name.lower().replace(" ", ".") + "@company.com"

# Output

print("\n--- Your Profile ---")

print("Full Name:", full\_name)

print("Generated Email:", email)

### **🧪 Example Usage:**

python email\_generator.py Gowtham S B

**Output:**

--- Your Profile ---

Full Name: Gowtham S B

## **🧵 8. Real-World Use Case**

Imagine you write a script to convert CSV to JSON.

* ❌ input() will ask file paths every time.
* ✅ sys.argv lets you pass them directly:

python converter.py input.csv output.json

## **🧩 9. Summary**

* Use input() for **learning and interactive apps**.
* Use sys.argv for **automated, scriptable, and production** code.
* Clean and professional scripts avoid input() and embrace argument parsing via sys.argv or argparse.

### **About the Author**

**Gowtham SB** is a **Data Engineering expert, educator,** **and content creator** with a passion for **big data technologies, as well as cloud and Gen AI** . With years of experience in the field, he has worked extensively with **cloud platforms, distributed systems, and data pipelines**, helping professionals and aspiring engineers master the art of data engineering.

Beyond his technical expertise, Gowtham is a **renowned mentor and speaker**, sharing his insights through engaging content on **YouTube and LinkedIn**. He has built one of the **largest Tamil Data Engineering communities**, guiding thousands of learners to excel in their careers.

Through his deep industry knowledge and hands-on approach, Gowtham continues to **bridge the gap between learning and real-world implementation**, empowering individuals to build **scalable, high-performance data solutions**.

𝐒𝐨𝐜𝐢𝐚𝐥𝐬

🎥𝐘𝐨𝐮𝐓𝐮𝐛𝐞 - https://www.youtube.com/@dataengineeringvideos

📸𝐈𝐧𝐬𝐭𝐚𝐠𝐫𝐚𝐦 - <https://instagram.com/dataengineeringtamil>

📸𝐈𝐧𝐬𝐭𝐚𝐠𝐫𝐚𝐦 - [https://instagram.com/](https://instagram.com/dataengineeringtamil)thedatatech.in

🤝𝐂𝐨𝐧𝐧𝐞𝐜𝐭 𝐟𝐨𝐫 𝟏:𝟏 - https://topmate.io/dataengineering/

💼𝐋𝐢𝐧𝐤𝐞𝐝𝐈𝐧 - https://www.linkedin.com/in/sbgowtham/

🌐𝐖𝐞𝐛𝐬𝐢𝐭𝐞 - https://codewithgowtham.blogspot.com

💻𝐆𝐢𝐭𝐇𝐮𝐛 - http://github.com/Gowthamdataengineer

💬𝐖𝐡𝐚𝐭𝐬 𝐀𝐩𝐩 - https://lnkd.in/g5JrHw8q

📧𝐄𝐦𝐚𝐢𝐥 - atozknowledge.com@gmail.com

📱𝐀𝐥𝐥 𝐌𝐲 𝐒𝐨𝐜𝐢𝐚𝐥𝐬 - <https://lnkd.in/gf8k3aCH>