Bhai, according to that video from CampusX on Generators in Python, here's the main difference between a normal function and a generator function, explained simply with Python examples. I'll keep it straightforward since you work on Python.

Quick Overview:

- **Normal Function**: Ye ek regular function hai jo poora execute hota hai jab call karte ho, ek single value return karta hai (using `return`), and then function khatam – no more state ya pausing.
- **Generator Function**: Ye special type ka function hai jo `yield` keyword use karta hai. Jab call karte ho, ye ek generator object return karta hai (iterator), values ko one-by-one yield karta hai on demand, and execution pause ho jata hai until next value chahiye. Memory super efficient hai for big data.

Key Differences (Step-by-Step):

1. **Execution Style**:

   - Normal: Poora code run ho jata hai ek baar mein. Example:

     ```
     Python

     def normal_func(n):
         result = []
         for i in range(n):
             result.append(i * 2)
         return result  # Returns full list at once

     print(normal_func(5))  # [0, 2, 4, 6, 8] - everything computed upfro
     ```

     Memory khata hai if n bada hai, kyunki full list bana deta hai.

   - Generator: Lazy evaluation – sirf tab compute karta hai jab value mangte ho (next() or loop mein). Example:

     ```
     Python

     def generator_func(n):
         for i in range(n):
             yield i * 2  # Yields one value, pauses here

     gen = generator_func(5)
     print(next(gen))  # 0
     print(next(gen))  # 2
     # And so on - doesn't compute all at once
     ```

2. **Return Value**:

   - Normal: Single value ya list/tuple etc. returns, function ends.

   - Generator: Generator object deta hai, jo iterable hai. Loop kar sakte ho:

     ```
     Python

     for val in generator_func(5):
         print(val)  # Prints 0,2,4,6,8 one by one
     ```

3. **Memory Use**:
    - Normal: Full data store karta hai memory mein (e.g., list of 100k items = ~800KB).
    - Generator: Sirf current state rakhta hai (~100 bytes), perfect for infinite sequences ya large files (jaise image reader mein ek-ek image load karo).
4. **State Management**:
    - Normal: No memory of previous calls – har call fresh start.
    - Generator: State remember karta hai (local variables), resume karta hai last yield se.

Why Generators Rock (From Video):

- Easy to implement iterators without boilerplate code.
- Memory save karta hai big loops ke liye.
- Infinite data handle kar sakte ho (e.g., endless even numbers).
- Chain kar sakte ho (ek generator ka output dusre mein feed).

Video mein examples hain jaise custom range banao ya Fibonacci square karo – try karo code_execution mein if you want demo. If you need code snippets to run, batao!