1. **Difference between a function and a method in Python:**
    - A function is an independent block of reusable code that performs a specific task and is called directly.
    - A method is similar to a function but is associated with an object and is called using the object's reference.
2. **Function arguments and parameters in Python:**
    - Parameters are placeholders defined in a function to accept values when it is called.
    - Arguments are the actual values passed to a function during execution.
3. **Ways to define and call a function in Python:**
    - Functions can be defined using the `def` keyword, followed by a name and parameters.
    - They can be called by using their name and providing necessary arguments.
    - Special function types include lambda functions, functions with variable-length arguments, and recursive functions.
4. **Purpose of the `return` statement in a Python function:**
    - The `return` statement allows a function to send back a value to the caller.
    - It helps in returning results from computations and terminating the function's execution.
5. **Iterators vs. Iterables in Python:**
    - An iterable is an object that can return its elements one at a time, such as lists, tuples, and strings.
    - An iterator is an object that follows the iterator protocol and retrieves elements sequentially using a special method.
6. **Concept of generators and how they are defined:**
    - A generator is a special type of iterator that produces values on demand without storing them in memory.
    - It uses the `yield` keyword to return values one at a time while maintaining its state between calls.
7. **Advantages of generators over regular functions:**
    - Generators improve memory efficiency by generating values on the fly instead of storing them.
    - They enable better performance for large data processing tasks.
    - They retain state between calls, reducing redundant computations.
8. **Lambda function in Python and its usage:**
    - A lambda function is an anonymous, single-expression function used for short and simple operations.
    - It is commonly used in functional programming, particularly in cases where a small function is needed temporarily.
9. **Purpose and usage of the `map()` function in Python:**
    - The `map()` function applies a given function to all elements in an iterable and returns a new iterable.
    - It is useful for performing transformations on a sequence without using explicit loops.
10. **Difference between `map()`, `reduce()`, and `filter()`:**

- `map()` applies a function to every element in an iterable and returns a transformed sequence.
- `filter()` selects elements based on a condition and returns a subset of the original sequence.
- `reduce()` applies a function cumulatively to elements, reducing them to a single result.