1. The difference between enclosing a list comprehension in square brackets and parentheses is that the former produces a list while the latter produces a generator object. For example, `[x\*\*2 for x in range(5)]` produces the list `[0, 1, 4, 9, 16]`, while `(x\*\*2 for x in range(5))` produces a generator object.

2. Generators are a type of iterator. While iterators allow you to iterate over a collection of items one at a time, generators enable you to define an iterator in a concise and efficient way using a function with a `yield` statement. Generators produce a sequence of values on-the-fly as you iterate over them, rather than constructing the entire sequence in memory upfront.

3. A function is a generator function if it contains a `yield` statement. The presence of a `yield` statement indicates that the function will return a generator object when called, which can be iterated over to produce a sequence of values.

4. The purpose of a `yield` statement is to generate a value in a generator function and pause the function's execution at that point, allowing the calling code to consume the generated value. When the generator is iterated over again, the function resumes its execution immediately after the `yield` statement and generates the next value, and so on until there are no more values to generate.

5. Both `map` calls and list comprehensions allow you to perform operations on a collection of items and generate a new sequence. The key difference is that `map` applies a function to each item in the sequence and returns an iterator over the resulting values, while a list comprehension generates a list of values by applying an expression to each item in the sequence. Another difference is that `map` may be more memory-efficient than a list comprehension when working with very large sequences, as it generates values on-the-fly rather than constructing an entire list in memory. However, list comprehensions may be more readable and concise in some cases, and they offer more flexibility in terms of the types of operations that can be performed on the input sequence.