6.8. `rlcompleter` — Completion function for GNU readline

Source code: Lib/rlcompleter.py

The `rlcompleter` module defines a completion function suitable for the `readline` module by completing valid Python identifiers and keywords.

When this module is imported on a Unix platform with the `readline` module available, an instance of the `Completer` class is automatically created and its `complete()` method is set as the `readline` completer.

Example:

```
>>> import rlcompleter
>>> import readline
>>> readline.parse_and_bind("tab: complete")
>>> readline. <TAB PRESSED>
readline.__doc__          readline.get_line_buffer(  readline.read_ini
readline.__file__         readline.insert_text(      readline.set_comp
readline.__name__         readline.parse_and_bind(
>>> readline.
```

The `rlcompleter` module is designed for use with Python's interactive mode. Unless Python is run with the `-S` option, the module is automatically imported and configured (see Readline configuration).

On platforms without `readline`, the `Completer` class defined by this module can still be used for custom purposes.

6.8.1. Completer Objects

Completer objects have the following method:

`Completer.`**`complete`**(*text*, *state*)

Return the *state*th completion for *text*.

If called for *text* that doesn't include a period character (`'.'`), it will complete from names currently defined in `__main__`, `builtins` and keywords (as defined by the `keyword` module).

If called for a dotted name, it will try to evaluate anything without obvious side-effects (functions will not be evaluated, but it can generate calls to `__getattr__()`) up to the last part, and find matches for the rest via the `dir()` function. Any exception raised during the evaluation of the expression is caught, silenced and `None` is returned.