Python File Methods:

Sure, here are 10 more examples of methods for working with files in Python:

1. `readlines()`: This method is used to read all the lines in a file and return them as a list. For example:

```
lines = file.readlines()
```

2. `writelines()`: This method is used to write a list of strings to a file. Each string is written to a new line. For example:

```
lines = ["Line 1\n", "Line 2\n", "Line 3\n"]
file.writelines(lines)
```

3. `seekable()`: This method is used to check if a file supports random access. For example:

```
if file.seekable():
    print("File supports random access")
```

4. `fileno()`: This method is used to get the file descriptor of a file object. For example:

```
fd = file.fileno()
```

5. `isatty()`: This method is used to check if a file is connected to a tty device. For example:

```
if file.isatty():
    print("File is connected to a tty device")
```

6. `name`: This attribute is used to get the name of the file. For example:

```
filename = file.name
```

7. `mode`: This attribute is used to get the mode in which the file was opened. For example:

```
filemode = file.mode
```

8. `closed`: This attribute is used to check if a file is closed. For example:

```
if file.closed:
    print("File is closed")
```

9. `readable()`: This method is used to check if a file is readable. For example:

```
if file.readable():
    print("File is readable")
```

10. `writable()`: This method is used to check if a file is writable. For example:

```
if file.writable():
    print("File is writable")
```

These methods and attributes provide a range of functionality for working with files in Python. By combining them with control structures, exception handling, and other features of the language, you can create robust and flexible programs that can read, write, and manipulate files in a variety of ways.