Reading Data From a File

Reading data from a file in Python is a common operation that involves using the `open()` function with the appropriate access mode ('r' for read, 'rb' for binary read, etc.) and then using methods like `read()`, `readline()`, or `readlines()` to retrieve the content. Here are examples demonstrating various ways to read data from a file:

1. **Read Mode ('r'):**
   This mode is used to read the contents of a file.

   ```python
   # Reading the entire content of a file
   with open('example.txt', 'r') as file:
       content = file.read()
       print(content)
   ```

   In this example, the entire content of 'example.txt' is read and printed.

2. **Reading Line by Line:**
   You can use a loop to read the file line by line using the `readline()` method.

   ```python
   # Reading a file line by line
   with open('example.txt', 'r') as file:
       line = file.readline()
       while line:
           print(line.strip())  # Strip removes newline characters
           line = file.readline()
   ```

   Alternatively, you can use a `for` loop:

   ~~~python
   # Reading a file line by line using a for loop
   with open('example.txt', 'r') as file:
       for line in

the file:
```python
           print(line.strip())
   ~~~

   This code reads and prints each line from 'example.txt' in the console.

3. **Reading All Lines as a List:**
   You can use the `readlines()` method to read all lines of a file into a list.

```python
   # Reading all lines of a file into a list
   with open('example.txt', 'r') as file:
       lines = file.readlines()
       for line in lines:
           print(line.strip())
   ```

   This code reads all lines of 'example.txt' into a list, and then each line is printed.

4. **Reading Binary Data:**
   For reading binary data, use binary read mode ('rb').

   ```python
   with open('output.bin', 'rb') as file:
       binary_data = file.read()
       # Process binary data as needed
   ```

   Here, 'output.bin' is read as binary data.

5. **Using Context Manager with `with` Statement:**
   The `with` statement is commonly used to ensure that the file is properly closed after reading.

   ```python
   with open('example.txt', 'r') as file:
       content = file.read()
       print(content)
   # File is automatically closed outside the 'with' block
   ```

   The `with` statement ensures that the file is closed, even if an exception occurs during the execution of the block.

Reading data from a file is an essential task in many Python applications, whether it's for data analysis, configuration, or simply retrieving information. Choose the appropriate read method based on whether you want to read the entire content, line by line, or as a list of lines. Always use the `with` statement to ensure proper handling of the file resources.