158. Read N Characters Given Read4 II - Call multiple times

Given a file and assume that you can only read the file using a given method `read4`, implement a method `read` to read *n* characters. **Your method `read` may be called multiple times.**

Method read4:

The API `read4` reads 4 consecutive characters from the file, then writes those characters into the buffer array `buf`.

The return value is the number of actual characters read.

Note that `read4()` has its own file pointer, much like `FILE *fp` in C.

Definition of read4:

```
    Parameter:  char[] buf
    Returns:    int

Note: buf[] is destination not source, the results from read4 will be copied to buf[]
```

Below is a high level example of how `read4` works:

```
File file("abcdefghijk"); // File is "abcdefghijk", initially file pointer (fp) points to 'a'
char[] buf = new char[4]; // Create buffer with enough space to store characters
read4(buf); // read4 returns 4. Now buf = "abcd", fp points to 'e'
read4(buf); // read4 returns 4. Now buf = "efgh", fp points to 'i'
read4(buf); // read4 returns 3. Now buf = "ijk", fp points to end of file
```

Method read:

By using the `read4` method, implement the method `read` that reads *n* characters from the file and store it in the buffer array `buf`. Consider that you **cannot** manipulate the file directly.

The return value is the number of actual characters read.

Definition of read:

```
    Parameters: char[] buf, int n
    Returns:    int

Note: buf[] is destination not source, you will need to write the results to buf[]
```

Example 1:

```
File file("abc");
Solution sol;
// Assume buf is allocated and guaranteed to have enough space for storing all characters from the
file.
sol.read(buf, 1); // After calling your read method, buf should contain "a". We read a total of 1
character from the file, so return 1.
sol.read(buf, 2); // Now buf should contain "bc". We read a total of 2 characters from the file, so
return 2.
sol.read(buf, 1); // We have reached the end of file, no more characters can be read. So return 0.
```

Example 2:

```
File file("abc");
Solution sol;
sol.read(buf, 4); // After calling your read method, buf should contain "abc". We read a total of 3
characters from the file, so return 3.
sol.read(buf, 1); // We have reached the end of file, no more characters can be read. So return 0.
```

Note:

1. Consider that you **cannot** manipulate the file directly, the file is only accesible for `read4` but **not** for `read`.
2. The `read` function may be called **multiple times**.
3. Please remember to **RESET** your class variables declared in Solution, as static/class variables are **persisted across multiple test cases**. Please see here for more details.
4. You may assume the destination buffer array, `buf`, is guaranteed to have enough space for storing *n* characters.
5. It is guaranteed that in a given test case the same buffer `buf` is called by `read`.