```
/**
 * The read4 API is defined in the parent class Reader4.
 *     int read4(char[] buf);
 */
public class Solution extends Reader4 {
    /**
     * @param buf Destination buffer
     * @param n   Number of characters to read
     * @return    The number of actual characters read
     */
    public int read(char[] buf, int n) {
        int idx = 0;
        while (idx < n) {
            char[] buffer = new char[4];
            int r = read4(buffer);
            int index = 0;
            while (index < r && idx < n) {
                buf[idx++] = buffer[index++];
            }
            if (r != 4) {
                break;
            }
        }
        return idx;
    }
}
```

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.

```
/**
 * The read4 API is defined in the parent class Reader4.
 *     int read4(char[] buf);
 */
public class Solution extends Reader4 {
    /**
     * @param buf Destination buffer
     * @param n   Number of characters to read
     * @return    The number of actual characters read
     */
    int curPtr = 0, idxPtr = 0;
    char[] buffer = new char[4];
    public int read(char[] buf, int n) {
        int ptr = 0;
        while (ptr < n) {
            if (curPtr == 0) {
                idxPtr = read4(buffer);
            }

            while (ptr < n && curPtr < idxPtr) {
                buf[ptr++] = buffer[curPtr++];
            }

            if (curPtr == idxPtr) {
                curPtr = 0;
            }

            if (idxPtr != 4) {
                break;
            }
        }
        return ptr;
    }
}
```