Read N Characters Given Read4 II - Call multiple times

The API: `int read4(char *buf)` reads 4 characters at a time from a file.

The return value is the actual number of characters read. For example, it returns 3 if there is only 3 characters left in the file.

By using the `read4` API, implement the function `int read(char *buf, int n)` that reads *n* characters from the file.

Note:
The `read` function may be called multiple times.

Solution 1

```
    private int buffPtr = 0;
    private int buffCnt = 0;
    private char[] buff = new char[4];
    public int read(char[] buf, int n) {
        int ptr = 0;
        while (ptr < n) {
            if (buffPtr == 0) {
                buffCnt = read4(buff);
            }
            if (buffCnt == 0) break;
            while (ptr < n && buffPtr < buffCnt) {
                buf[ptr++] = buff[buffPtr++];
            }
            if (buffPtr >= buffCnt) buffPtr = 0;
        }
        return ptr;
    }
```

I used buffer pointer (buffPtr) and buffer Counter (buffCnt) to store the data received in previous calls. In the while loop, if buffPtr reaches current buffCnt, it will be set as zero to be ready to read new data.

written by totalheap original link here

Solution 2

The key is to store memorized variable in the class level and remember offset position and remaining number of elements.

```
/* The read4 API is defined in the parent class Reader4.
      int read4(char[] buf); */

public class Solution extends Reader4 {

    private int offSet = 0;
    private int remaining = 0;
    private boolean isEndOfFile = false;
    private char[] buffer = new char[4];

    /**
     * @param buf Destination buffer
     * @param n   Maximum number of characters to read
     * @return    The number of characters read
     */
    public int read(char[] buf, int n) {
        int readBytes = 0;
        while (readBytes < n && (remaining != 0 || !isEndOfFile)) {
            int readSize = 0;
            if (remaining != 0) {
                readSize = remaining;
            } else {
                offSet = 0;
                readSize = read4(buffer);
                if (readSize != 4) {
                    isEndOfFile = true;
                }
            }
            int length = Math.min(n - readBytes, readSize);
            for (int i= offSet; i<offSet + length; i++) {
                buf[readBytes++] = buffer[i];
            }
            remaining = readSize - length;
            if (remaining != 0) {
                offSet += length;
            }
        }
        return readBytes;
    }
}
```

written by richilee original link here

Solution 3

```
class Solution:
# @param buf, Destination buffer (a list of characters)
# @param n,   Maximum number of characters to read (an integer)
# @return     The number of characters read (an integer)
def __init__(self):
    self.queue = []

def read(self, buf, n):
    idx = 0
    while True:
        buf4 = [""]*4
        l = read4(buf4)
        self.queue.extend(buf4)
        curr = min(len(self.queue), n-idx)
        for i in xrange(curr):
            buf[idx] = self.queue.pop(0)
            idx+=1
        if curr == 0:
            break
    return idx
```

written by ycsung original link here