```
/* The following code example is taken from the book
 * "The C++ Standard Library - A Tutorial and Reference, 2nd Edition"
 * by Nicolai M. Josuttis, Addison-Wesley, 2012
 *
 * (C) Copyright Nicolai M. Josuttis 2012.
 * Permission to copy, use, modify, sell and distribute this software
 * is granted provided this copyright notice appears in all copies.
 * This software is provided "as is" without express or implied
 * warranty, and with no claim as to its suitability for any purpose.
 */
#include <cstdio>
#include <cstring>
#include <streambuf>

// for read():
#ifdef _MSC_VER
# include <io.h>
#else
# include <unistd.h>
#endif

class inbuf : public std::streambuf {

  protected:
    // data buffer:
    // - at most, four characters in putback area plus
    // - at most, six characters in ordinary read buffer
    static const int bufferSize = 10;    // size of the data buffer
    char buffer[bufferSize];             // data buffer

  public:
    // constructor
    // - initialize empty data buffer
    // - no putback area
    // => force underflow()
    inbuf() {
        setg (buffer+4,     // beginning of putback area
              buffer+4,     // read position
              buffer+4);    // end position
    }

  protected:
    // insert new characters into the buffer
    virtual int_type underflow () {
        // is read position before end of buffer?
        if (gptr() < egptr()) {
            return traits_type::to_int_type(*gptr());
        }

        // process size of putback area
        // - use number of characters read
        // - but at most four
        int numPutback;
        numPutback = gptr() - eback();
        if (numPutback > 4) {
            numPutback = 4;
```

```
        }

        // copy up to four characters previously read into
        // the putback buffer (area of first four characters)
        std::memmove (buffer+(4-numPutback), gptr()-numPutback,
                      numPutback);

        // read new characters
        int num;
        num = read (0, buffer+4, bufferSize-4);
        if (num <= 0) {
            // ERROR or EOF
            return EOF;
        }

        // reset buffer pointers
        setg (buffer+(4-numPutback),   // beginning of putback area
              buffer+4,                // read position
              buffer+4+num);           // end of buffer

        // return next character
        return traits_type::to_int_type(*gptr());
    }
};
```