序号。(因此 SendBase－1 是指接收方已正确按序接收到的数据的最后一个字节的序号。) 如前面指出的那样，TCP 采用累积确认，所以 y 确认了字节编号在 y 之前的所有字节都已经收到。如果 y > SendBase，则该 ACK 是在确认一个或多个先前未被确认的报文段。因此发送方更新它的 SendBase 变量；如果当前有未被确认的报文段，TCP 还要重新启动定时器。

```
/* 假设发送方不受TCP流量和拥塞控制的限制，来自上层的数据长度小于MSS，且数据传送只在一个
方向进行。*/

NextSeqNum=InitialSeqNumber
SendBase=InitialSeqNumber

loop (forever) {
    switch(event)

        event: data received from application above
            create TCP segment with sequence number NextSeqNum
            if (timer currently not running)
                start timer
            pass segment to IP
            NextSeqNum=NextSeqNum+length(data)
            break;

        event: timer timeout
            retransmit not-yet-acknowledged segment with
                smallest sequence number
            start timer
            break;

        event: ACK received, with ACK field value of y
            if (y > SendBase) {
                SendBase=y
                if (there are currently any not-yet-acknowledged segments)
                    start timer
                }
            break;

    } /* forever 循环结束 */
```

图 3-33 简化的 TCP 发送方

1. 一些有趣的情况

我们刚刚描述了一个关于 TCP 如何提供可靠数据传输的高度简化的版本。但即使这种高度简化的版本，仍然存在着许多微妙之处。为了较好地感受该协议的工作过程，我们来看几种简单情况。图 3-34 描述了第一种情况，主机 A 向主机 B 发送一个报文段。假设该报文段的序号是 92，而且包含 8 字节数据。在发出该报文段之后，主机 A 等待一个来自主机 B 的确认号为 100 的报文段。虽然 A 发出的报文段在主机 B 上被收到，但从主机 B 发往主机 A 的确认报文丢失了。在这种情况下，超时事件就会发生，主机 A 会重传相同的报文段。当然，当主机 B 收到该重传的报文段时，它将通过序号发现该报文段包含了早已收到的数据。因此，主机 B 中的 TCP 将丢弃该重传的报文段中的这些字节。

在第二种情况中，如图 3-35 所示，主机 A 连续发回了两个报文段。第一个报文段序号是 92，包含 8 字节数据；第二个报文段序号是 100，包含 20 字节数据。假设两个报文段都完好无损地到达主机 B，并且主机 B 为每一个报文段分别发送一个确认。第一个确认报