Enter a line of text: A short line.

The line has 14 characters, including 2 blanks
<4 lemon:˜/Examples > gcc -c -mbig-endian -Xassembler -als example31.s

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Value</th>
<th>A/R</th>
<th>L/G</th>
<th>Definition</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>.text+00000000</td>
<td>REL</td>
<td>G</td>
<td>line 9</td>
<td>.text+30</td>
</tr>
<tr>
<td>loop</td>
<td>.text+00000014</td>
<td>REL</td>
<td>L</td>
<td>line 17</td>
<td>.text+30</td>
</tr>
<tr>
<td>if</td>
<td>.text+0000001c</td>
<td>REL</td>
<td>L</td>
<td>line 20</td>
<td>.text+20</td>
</tr>
<tr>
<td>then</td>
<td>.text+00000024</td>
<td>REL</td>
<td>L</td>
<td>line 23</td>
<td>.text+20</td>
</tr>
<tr>
<td>endif</td>
<td>.text+00000028</td>
<td>REL</td>
<td>L</td>
<td>line 25</td>
<td>.text+30</td>
</tr>
<tr>
<td>end</td>
<td>.text+00000034</td>
<td>REL</td>
<td>L</td>
<td>line 30</td>
<td>.text+20</td>
</tr>
<tr>
<td>fmt1</td>
<td>.text+0000004c</td>
<td>REL</td>
<td>L</td>
<td>line 38</td>
<td>.text+04</td>
</tr>
<tr>
<td>fmt2</td>
<td>.text+00000064</td>
<td>REL</td>
<td>L</td>
<td>line 39</td>
<td>.text+34</td>
</tr>
</tbody>
</table>

/* Example #31 -- Assembler processing */

.global main

.text

main:   push    {lr}

ldr     r0, =fmt1 @ First arg -- address of format
bl      printf @ Prompt the user to enter a line

mov     r4, #0 @ Initialize character count
mov     r5, #0 @ Initialize blank count

add     r4, r4, #1 @ Increment character count
cmp     r0, #0x20 @ Compare return value to blank
bne     endif @ If blank not found, skip inc

add     r5, r5, #1 @ Increment blank count

add     r5, r5, #1 @ Increment blank count

cmp     r0, #0x0a @ Compare return value to newline
beq     end @ If newline found, exit loop

loop:  bl      getchar @ Read one character (in r0)

add     r4, r4, #1 @ Increment character count

endif: cmp     r0, #0x20 @ Compare return value to blank
bne     endif @ If blank not found, skip inc

add     r5, r5, #1 @ Increment blank count

end:   ldr     r0, =fmt2 @ First arg  -- address of format
mov     r1, r4 @ Second arg -- character count
mov     r2, r5 @ Third arg -- blank count
bl      printf @ Display the two counts

loop:  b       loop @ Branch to top of loop

end:   ldr     r0, =fmt2 @ First arg  -- address of format
mov     r1, r4 @ Second arg -- character count
mov     r2, r5 @ Third arg -- blank count
bl      printf @ Display the two counts

pop     (lr)

bx      lr

fmt1:  .asciz  "\nEnter a line of text: "

fmt2:  .asciz  "\nThe line has %d characters, including ...

