Source File: `lab26.asm`
Input: Standard Input
Output: Standard Output
Value: 1

Write an assembly language function that receives a signed 32-bit integer and displays its internal binary representation. Use the `bt` instruction to access the bits. Using any shift or division instruction is prohibited. A description of the function as well as client code for testing your implementation is shown in Figure 1, and a sample execution sequence is shown in Figure 2. To use the `Makefile` as distributed in class, add a target of `lab26` to `targets2AsmFiles`.

```
1  [list -]
2  %INCLUDE "Along32.inc"
3  %INCLUDE "Macros_Along.inc"
4  [list +]
5
6  ;-----------------------------------------------------------------------
7  extern  PrintBinary
8  ; HLL prototype: void PrintBinary(int n);
9  ; Prints the internal binary representation of n
10 ; Receives: EAX = signed 32-bit integer
11 ; Returns:  nothing
12 ;-----------------------------------------------------------------------
13
14 SECTION .data
15 hrule   times   51 db ('-')
16         db      10,0
17 spacer1 times   2 db ' '
18         db      0
19 spacer2 times   4 db ' '
20         db      0
21 header  times   2 db ' '
22         db      '  Decimal  '
23         times   17 db ' '
24         db      'Binary',10,0
25 ten     dd      10
26
27 SECTION .bss
28 h       resd    1
29 num     resd    1
30 width   resd    1
31
32 SECTION .text
33         global  _start
34 _start:
35         call    ReadDec                 ; read an unsigned integer
36         mov     [h],eax                 ; move the integer to n
37
38         mov     edx,hrule               ; print the table header
39         call    WriteString
```

Figure 1. `/usr/local/3304/src/lab26main.asm` (Part 1 of 2)

```
40          mov     edx,header
41          call    WriteString
42          mov     edx,hrule
43          call    WriteString
44  .L0:
45          cmp     dword [h],0             ; while h >= 0 do
46          je      .L5
47          call    ReadInt                 ; read a signed 32-bit integer
48          mov     dword [num], eax        ; save a copy in num
49          mov     edx,spacer1
50          call    WriteString
51                                          ; determine the width of the input num
52          mov     dword [width],1
53          mov     eax,[num]
54  .L1:
55          cdq                             ; convert from dword to qword
56          idiv    dword [ten]             ; signed division by 10
57          cmp     eax,0                   ; if the quotient is 0, we're done
58          je      .L2
59          inc     dword [width]           ; else increment the width
60          jmp     .L1
61  .L2:
62          mov     ecx,10                  ; insert enough spaces to right-justify
63          sub     ecx,[width]             ; num
64  .L3:
65          cmp     ecx,0
66          je      .L4
67          mov     al,' '
68          call    WriteChar
69          dec     ecx
70          jmp    .L3
71  .L4:
72          mov     eax,[num]
73          call    WriteInt
74          mov     edx,spacer2
75          call    WriteString
76          call    PrintBinary
77          mov     al,10
78          call    WriteChar
79          dec     dword [h]
80          jmp     .L0                     ; end while
81  .L5:
82          mov     edx,hrule               ; print the table footer
83          call    WriteString
84          Exit    {0}
```

Figure 1. `/usr/local/3304/src/lab26main.asm` (Part 2 of 2)

```
newuser@csunix ~/3304/26> cp /usr/local/3304/data/26/* .
newuser@csunix ~/3304/26> cp /usr/local/3304/src/Makefile .
newuser@csunix ~/3304/26> cp /usr/local/3304/src/lab26main.asm .
newuser@csunix ~/3304/26> touch lab26.asm
newuser@csunix ~/3304/26> make lab26
nasm -f elf32  -l lab26main.lst -o lab26main.o lab26main.asm -I/usr/local/3304/include/ -I.
nasm -f elf32  -l lab26.lst -o lab26.o lab26.asm -I/usr/local/3304/include/ -I.
ld -m elf_i386 --dynamic-linker /lib/ld-linux.so.2 -o lab26 lab26main.o lab26.o \
/usr/local/3304/src/Along32.o -lc
newuser@csunix ~/3304/26> ../irvine_test.sh lab26 01.dat
-------------------------------------------------
    Decimal                   Binary
-------------------------------------------------
           +0    00000000000000000000000000000000
           +1    00000000000000000000000000000001
           -1    11111111111111111111111111111111
           +2    00000000000000000000000000000010
           -2    11111111111111111111111111111110
           +3    00000000000000000000000000000011
           -3    11111111111111111111111111111101
          +12    00000000000000000000000000001100
          -12    11111111111111111111111111110100
         +123    00000000000000000000000001111011
         -123    11111111111111111111111110000101
        +1234    00000000000000000000010011010010
        -1234    11111111111111111111101100101110
       +12345    00000000000000000011000000111001
       -12345    11111111111111111100111111000111
      +123456    00000000000000011110001001000000
      -123456    11111111111111100001110111000000
     +1234567    00000000001001011010110000111
     -1234567    11111111111011010010100101111001
    +12345678    00000000101111000110000101001110
    -12345678    11111111010000111001111010110010
   +123456789    00000111010110111100110100010101
   -123456789    11111000101001000011001011101011
  +2147483647    01111111111111111111111111111111
  -2147483647    10000000000000000000000000000001
  -2147483648    10000000000000000000000000000000
-------------------------------------------------
newuser@csunix ~/3304/26> ../irvine_test.sh lab26 01.dat > my.out
newuser@csunix ~/3304/26> diff 01.out my.out
newuser@csunix ~/3304/26>
```

Figure 2. Commands to Assemble, Link, & Run Lab 26