```
section .data

      nline db 10,10
      nline_len equ $-nline

      arr dd -11111111H, 22222222H, 33333333H, -44444444H, -55555555H
      arr_size equ 5

      pmsg db     10,"The no. of Positive elements in 32-bit array:"
      pmsg_len equ $-pmsg

      nmsg db 10,10,"The no. of Negative elements in 32-bit array:"
      nmsg_len equ $-nmsg

section .bss

      p_count resq 01
      n_count resq 01
      dnumbuff resb 02

      %macro display 2
            mov rax,01
            mov rdi,01
            mov rsi,%1
            mov rdx,%2
            syscall
      %endmacro


section .text

global _start

_start:
      mov esi, arr
      mov ecx,5   ;Arraay counter i.e.5
      mov ebx,0   ; counter for +ve nos
      mov edx,0   ; counter for -ve nos

next_num:
      mov eax,[esi]     ; take no. in RAX
      rcl eax,1   ; rotate left 1 bit to check for sign bit
      jc negative

positive:
      inc ebx           ; no carry, so no. is +ve
      jmp next

negative:
      inc edx     ; carry, so no. is -ve

next:
      add esi,4   ; 32 bit nos i.e. 4 bytes
      loop next_num

      mov [p_count], ebx      ; store positive count
      mov [n_count], edx      ; store negative count

      display pmsg, pmsg_len
      mov ebx,[p_count] ; load value of p_count in rax
      call disp8_proc   ; display p_count

      display nmsg, nmsg_len
      mov ebx,[n_count] ; load value of n_count in rax
      call disp8_proc         ; display n_count

      display nline, nline_len

exit:
      mov rax,60        ;Exit
      mov rbx,00
      syscall




disp8_proc:
      mov edi,dnumbuff  ;point edi to buffer
      mov ecx,02        ;load number of digits to display

dispup1:
      rol bl,4    ;rotate number left by four bits
      mov dl,bl   ;move lower byte in dl
      and dl,0fh  ;mask upper digit of byte in dl
      add dl,30h  ;add 30h to calculate ASCII code
      cmp dl,39h  ;compare with 39h
      jbe dispskip1     ;if less than 39h skip adding 07 more
      add dl,07h  ;else add 07

dispskip1:
      mov [edi],dl      ;store ASCII code in buffer
      inc edi           ;point to next byte
      loop dispup1      ;decrement the count of digits to display
            ;if not zero jump to repeat

      display dnumbuff,2
      ret
```