```
Exemplos: eltd03_aula05 - 2024_1S
;---ex5_1a - instr. ASR 2024_1S
        export  __main
;===== diretiva area - dados (sram)
        area    dds1, data, readwrite
vt1     space   4
vt2     space   4
;=== diretiva area - prog. (flash)
        area    m_prog, code, readonly
__main
        ldr     r0,=vt1-1       ;pointer vt1
        mov     r1,#4   ;counter
pk0     ldrsb   r2,[r0,#1]!;Index.pré-incr.
        asr     r3,r2,#1
        strb    r3,[r0,#(vt2-vt1)]
        subs    r1,#1
        bne     pk0
        b       __main  ;prox.elem.
        end

;************************
;---Ex5_1c - instr. LSRS
        export  __main
;===== diretiva area - dados (sram)
        area    dds1, data, readwrite
vt1     space   4       ;vetor origem
vt2     space   4       ;vetor par
vt3     space   4       ;vetor ímpar
;=== diretiva area - prog. (flash)
        area    m_prog, code, readonly
__main
        ldr     r0,=vt1-1       ;pointer vt1
        mov     r1,#4   ;counter
pk0     ldrsb   r2,[r0,#1]!;Index.pré-incr.
        lsrs    r3,r2,#1
        bcs     imp     ;
par     strb    r3,[r0,#vt2-vt1]
        b       pk1
imp     strb    r3,[r0,#vt3-vt1]
pk1     subs    r1,#1
        bne     pk0
        b       __main  ;prox.elem.
        end

;************************
;---Ex5_1e - instr. LSL
        export  __main
nb      equ     4
;===== diretiva area - dados (sram)
        area    dds1, data, readwrite
vt1     space   nb
vt2     space   nb
;=== diretiva area - prog. (flash)
        area    m_prog, code, readonly
```

```
__main
        ldr     r0,=vt1+nb      ;pointer vt1
        mov     r1,#nb  ;counter
pk0     ldrsb   r2,[r0,#-1]! ;Index.pré-decr.
        lsl     r3,r2,#1
        strb    r3,[r0,#vt2-vt1]
        subs    r1,#1
        bne     pk0
        b       __main  ;prox.elem.
        end

;************************
;---Ex.5_1g - instr. RORS
        export  __main
nb      equ     4
;===== diretiva area - dados (sram)
        area    dds1, data, readwrite
vt1     space   nb
vt2     space   nb
vt3     space   nb
;=== diretiva area - prog. (flash)
        area    m_prog, code, readonly
__main
        ldr     r0,=vt1+nb      ;pointer vt1
        mov     r1,#nb  ;counter
pk0     ldrsb   r2,[r0,#-1]! ;Index.pré-decr.
        rors    r3,r2,#1
        bcs     imp
par     strb    r2,[r0,#vt2-vt1]
        b       pk1
imp     strb    r2,[r0,#vt3-vt1]
pk1     subs    r1,#1
        bne     pk0
        b       __main  ;prox.elem.
        end

;************************
;---Ex5_2a - flag V (vp+vp)
        export  __main
;===== diretiva area - dados (sram)
        area    dds1, data, readwrite
g1      space   1
h1      space   1
r1b     space   1
r2b     space   2
;=== diretiva area - prog. (flash)
        area    m_prog, code, readonly
__main
pk0     ldr     r0,=g1  ;pointer g1
        ldrb    r1,[r0] ;ler g1
        add     r0,#1
        ldrb    r2,[r0] ;ler h1
        lsl     r1,r1,#24
        lsl     r2,r2,#24
```

```
        adds    r3,r2,r1
        bvs     pk2
pk1     add     r0,#1   ;apontar r1b
        lsr     r3,r3,#24
        strb    r3,[r0] ;salvar em r1b
        b       pk0     ;cicl.
pk2     add     r0,#2   ;apontar r2b
        lsr     r3,r3,#24
        strh    r3,[r0]
        b       pk0     ;prox.elem.
        end

;************************
;---ex1_f - End.Index. offset imediato(#)
;com pré-incremento
        export  __main
;===== diretiva area - dados (sram)
        area    dds1, data, readwrite
vt1     space   8
vtz     dcw     &39af,&b287
vt2     space   8
;=== diretiva area - prog. (flash)
        area    m_prog, code, readonly
__main
        ldr     r0,=(vt1-1)     ;load pointer vt1
        mov     r1,#8           ;counter
pk0     ldrb    r2,[r0,#1]!     ;ler vt1(i)
        strb    r2,[r0,#(vt2-vt1)]      ;salvar em vt2(i)
;       add     r0,#1           ;inc.pointer DESNECESSÁRIO!!!
        subs    r1,#1           ;decr.counter
        bne     pk0             ;prox.elem.
        b       __main          ;
        end

;************************
;---Ex. 1g - End.index.offset imediato
;(auto) pós-incremento
        export  __main
;===== equates
;
;===== diretiva area - dados (sram)
        area    dds1, data, readwrite
vt1     space   8
vtz     dcw     &39af,&b287
vtw     dcb     &a7,&3b,&c5
vt2     space   8
;=== diretiva area - prog. (flash)
        area    m_prog, code, readonly
__main
        ldr     r0,=vt2         ;load pointer vt2
        mov     r1,#8           ;counter
pk0     ldrb    r2,[r0,#(vt1-vt2)]      ;ler vt1(i)
        strb    r2,[r0],#1      ;salvar em vt2(i)
;       add     r0,#1           ;inc.pointer
```

```
subs    r1,#1           ;decr.counter
bne     pk0             ;prox.elem.
b       __main          ;
end
```