Errata för tryckning 6.3:

2023-01-28

Senaste tryckning är 6:3,

Uppgift 1.7, flera fel i texten, rätta enligt:

UPPGIFT 1.7

Vi har deklarationerna:

```
int i;
unsigned short usvec[64];
signed short svec[72];
```

Koda tilldelningssatsen: `svec[14] = usvec[i];` genom att komplettera följande;

```
        LDR       R0,=i          @ R0 <- &i
                                 @ R0 <- i
                                 @ R0 <- i*sizeof(short)
                                 @ R0 <- &usvec[0]
                                 @ R1 <- usvec[i]
                                 @ R0 <- &svec[0]
                                 @ R1 -> svec[14]
        .ALIGN
usvec:                           @ usvec[64]
                                 @ svec[72]
                                 @ i
```

Uppgift 1.24, fel ”mall” ska vara:

UPPGIFT 1.24

Vi har deklarationerna

```
short a,b; int x,i,j;  int ai[32][8];
```

på "toppnivå".

a) Visa hur deklarationerna kodas

Visa hur tilldelningarna:

b) `x = (a+b) * (a-b);`

c) `ai[i][j] = x;`

kodas i ARMv6 assemblerspråk..

a)

```
                        @ a
                        @ b
                        @ x
                        @ i
                        @ j
                        @ ai[32][8]
```

Uppgift 1.41, fel i ”mall” ska ”R5<-y”:

```
g:                      @ LR återanvänds, spill x och y
                        @ R4<- x
                        @ R5<- y
                        @ R0<- next()
                        @ R0<- next()-1
```

FACIT:

Flera fel i ursprunglig lösning, rätta enligt:

LÖSNINGSFÖRSLAG UPPGIFT 1.7

```
        LDR     R0,=i           @ R0 <- &i
        LDR     R0,[R0]         @ R0 <- i
        LSL     R0,R0,#1        @ R0 <- i*sizeof(short)
        LDR     R0,=usvec       @ R0 <- &usvec[0]
        LDRH    R1,[R1,R0]      @ R1 <- usvec[i]
        LDR     R0,=svec        @ R0 <- &svec[0]
        STRH    R1,[R0,#2*14]   @ R1 -> svec[14]
        .ALIGN
usvec:  .SPACE  128             @ usvec[64]
svec:   .SPACE  144             @ svec[72]
i:      .SPACE  4               @ i
```

LÖSNINGSFÖRSLAG UPPGIFT 1.8

```
        LDR     R0,=j           @ R0 <- &j
        LDR     R0,[R0]         @ R0 <- j
        LDR     R1,=cvec        @ R1 <- &cvec[0]
        LDRSB   R0,[R0,R1]      @ R0 <- cvec[j]
        LDR     R1,=i           @ R1 <- &i
        LDR     R1,[R1]         @ R1 <- i
        LSL     R1,R1,#1        @ R1 <- i*2
        LDR     R2,=svec        @ R2 <- &svec[0]
        STRH    R0,[R1,R2]      @ R0 -> svec[i]
        .ALIGN
cvec:   .SPACE  64              @ cvec[64]
svec:   .SPACE  176             @ svec[88]
i:      .SPACE  4               @ i
j:      .SPACE  4               @ j
```

LÖSNINGSFÖRSLAG UPPGIFT 1.18

```
        LDR     R0,=i           @ R0 <- &i
        LDRB    R0,[R0]         @ R0 <- (unsigned char) i
        SXTB    R0,R0           @ R0 <- (int) i
        ADD     R0,#16          @ R0 <- (i+16)
        LDR     R1,=j           @ R1 <- &j
        LDRB    R1,[R1]         @ R1 <- (unsigned char) j
        SXTB    R1,R1           @ R1 <- (int) j
        SUB     R0,R0,R1        @ R0 <- (i+16)-j
        .ALIGN
i:      .SPACE  2               @ i
j:      .SPACE  2               @ j
```

LÖSNINGSFÖRSLAG UPPGIFT 1.19

```
        LDR     R0,=b           @ R0 <- &b
        LDR     R0,[R0]         @ R0 <- b
        LDR     R1,=c           @ R1 <- &c
        LDR     R1,[R1]         @ R1 <- c
        SUB     R0,R0,R1        @ R0 <-(b-c)
        LDR     R2,=a           @ R2 <- &a
        LDR     R1,[R2]         @ R1 <- a
        ADD     R1,R1,R0        @ R1 <- a+(b-c)
        STR     R1,[R2]         @ a+(b-c)] -> a
        .ALIGN
x:      .SPACE  4               @ x
y:      .SPACE  4               @ y
z:      .SPACE  4               @ z
```

LÖSNINGSFÖRSLAG UPPGIFT 1.22

```
        LDR         R0,=y           @ R0 <- &y
        LDR         R0,[R0]         @ R0 <- y
        LDR         R1,=z           @ R1 <- &z
        LDR         R1,[R1]         @ R1 <- z
        ADD         R0,R0,R1        @ R0 <- y+z
        LDR         R1,=x           @ R1 <- &x
        LDR         R1,[R1]         @ R1 <- x
        MUL         R0,R1           @ R0 <- x*(y+z)
        LDR         R2,=v           @ R2 <- &v
        LSL         R1,R1,#2        @ R1 <- x*sizeof(int)
        ADD         R1,R2,R1        @ R1 <- &v+ x*sizeof(int)
        LDR         R1,[R1]         @ R1 <- v[x]
        SUB         R0,R0,R1        @ R0 <- x*(y+z)-v[x]
```

LÖSNINGSFÖRSLAG UPPGIFT 1.24

a)

```
a:      .SPACE      2               @ a
b:      .SPACE      2               @ b
x:      .SPACE      4               @ x
i:      .SPACE      4               @ i
j:      .SPACE      4               @ j
ai:     .SPACE      32*8*4          @ ai[32][8]
```

c)

```
        LDR         R0,=x           @ R0<- &x
        LDR         R0,[R0]         @ R0<- x
        LDR         R1,=i           @ R1<- &i
        LDR         R1,[R0]         @ R1<- i
        LDR         R2,=j           @ R2<- &j
        LDR         R2,[R0]         @ R2<- j
        LSL         R1,R1,#3        @ R1<- i*8
        ADD         R1,R1,R2        @ R1<- (i*8)+j
        LSL         R1,R1,#2        @ R1<- ((i*8)+j)*sizeof(int)
        LDR         R2,=ai          @ R2<- &ai
        ADD         R1,R2,R1        @ R1<- &ai+((i*8)+j)*sizeof(int)
        STR         R0,[R1]         @ x->ai[i,j]
```

LÖSNINGSFÖRSLAG UPPGIFT 1.37

```
@ int f( int p )
f:          PUSH        {LR}            @ LR återanvänds
            LSL         R0,R0,#2        @ R0 <- p<<2
            BL          reval           @ reval(p<<2)
            CMP         R0,#0           @ (rv > 0)?
```