From last time

Explain how integer values are represented on the ARM. (3 marks)
Usually in 32 bit words
Base 2, to simplify conversion and arithmetic
Top bit = sign (0=+ve, 1=-ve) (modulo, 2s-complement)

Explain why Pseudo Instructions are used in ARM assembly language. (2 marks)
some simple ideas e.g. “load a literal into a register” or “add a -ve literal” are non-trivial to write, so assembler translates them for us e.g. “ADR ... address” or “SUB ... +ve literal”

Explain why only certain literal values can be used in ARM instructions. (2 marks)
An arithmetic instruction only has about 12 bits to describe a literal, so only around 4K different values out of 4G \(2^{32}\) possible.
Question

\[ a = (b + c) \times (d + e) \times (b + d); \]

LDR R0, b
LDR R1, c
ADD R1, R0, R1 ; R1=(b+c) but keep R0=b for later
LDR R2, d
LDR R3, e
ADD R3, R2, R3 ; R3=(d+e) but keep R2=d for later
MUL R1, R1, R3 ; R1=(b+c)*(d+e)
ADD R0, R0, R2 ; R0=(b+d)
MUL R0, R1, R0 ; R0=(b+d)*(b+c)*(d+e)
STR R0, a

Rearrange to only need 3 registers?
\[ a = (b + c) \times (b + d) \times (d + e); \]
Question

ARM code for: \[ a = (2 - b) \times (c - 2); \]

LDR R0, b
RSB R0, R0, #2
LDR R1, c
SUB R1, R1, #2
MUL R0, R0, R1
STR R0, a
Question

ARM code for: \( a = (b \times 2) + (c \times -2); \)

LDR R0, b
MOV R1, #2
MUL R0, R0, R1
LDR R1, c
MOV R2, #-2
MUL R1, R1, R2 \{ MLA R0, R1, R2, R0 \}
ADD R0, R0, R1
STR R0, a