**Functions:**

May modify R0 – R3

Must preserve R4 – R8

Input parameters will be in R0 – R3, assigned from left-to-right.

Function return value in R0 (or R0,R1 if 64 bits).

uint32\_t cm2in(uint32\_t cm)

{

return (1690932006 \* cm) >> 32 ;

}

export cm2in

cm2in: LDR R1,=16990932006 ; (2^32)/2.54

UMULL R1,R0,R0,R1 ; discard R1, result in R0

BX LR

int32\_t discriminant(int32\_t a, int32\_t b, int32\_t c)

{

return b\*b – 4\*a\*c ;

}

export discriminant

discriminant: MUL R1,R1,R1 ; R1 🡨 b\*b

MUL R0,R0,R2 ; R0 🡨 a\*c

SUB R0,R1,R0,LSL#2 ; R0 🡨 b\*b – 4\*a\*c

BX LR

uint32\_t power(uint32\_t x, uint32\_t exp)

{

uint32\_t result = 1 ;

while (exp > 0) { result \*= x ; exp = exp – 1 ; }

return result ;

}

export power

power: ; R0 = x, R1 = exp

LDR R2,=1 ; result = 1

again: CBZ R1,done ; exp > 0 ?

MUL R2,R2,R0 ; result \*= x

SUB R1,R1,#1 ; exp = exp - 1

B again

done: BX LR

uint32\_t factorial(uint32\_t n)

{

If (n == 0) return 1 ;

return n\*factorial(n-1) ;

}

export factorial

factorial: ; R0 = n

PUSH {LR,R4} ; preserve LR and R4

MOV R4,R0 ; R4 🡨 n (copy it to a safe place)

LDR R0,=1 ; assume n == 0 and setup return value

CBZ R4,done ; if (n == 0) return 1

SUB R0,R4,#1 ; R0 🡨 n - 1

BL factorial ; modifies LR, R0 🡨 factorial(n-1)

MUL R0,R0,R4 ; R0 🡨 n \* factorial(n-1)

done: POP {LR,R4} ; restore LR and R4

BX LR