CS2208 **JING XU**
ASS #5 **250942790**

PART ONE: SOURCE COED

```
            AREA a5q1, CODE, READONLY
            ENTRY
X           EQU 4                       ;define x
N           EQU 2                       ;define n

MAIN        ADR sp, STACK               ;define the stack
            MOV r0, #X                  ;pass the parameter x to r0
            MOV r1, #N                  ;pass the parameter n to r1
            STMFD sp!, {r0-r1}          ;push x and n parameters in the stack
            SUB sp, sp, #4              ;reserve above the parameters for return value
            BL POWER                    ;call function power

            LDR r2, [sp], #12           ;load return value to r2, clean stack
            ADR r3, RESULT              ;get the address of RESULT
            STR r2, [r3]                ;store return value of POWER in RESULT address

END         B END                       ;Infinite loop to end the program

POWER       STMFD sp!, {r0-r1, fp, lr}  ;push registers to be modified, as well as fp and lr
            MOV fp, sp                  ;set current frame pointer to the top of the stack
            SUB sp, sp, #4              ;allocate space in the frame for local variable y
            LDR r1, [fp, #24]           ;load parameter n passed
            CMP r1, #0                  ;check if n==0
            MOVEQ r1, #1                ;if n==0, return 1
            BEQ RETURN                  ;jump to RETURN

            LDR r0, [fp, #20]           ;if n isn't 0, load parameter x
            TST r1, #1                  ;test last bit in n to determine if even or odd
            BNE ODD                     ;if n odd, jump to ODD
            B EVEN                      ;else, jump to EVEN

ODD         SUB r1, r1, #1              ;decrement n
            STMFD sp!, {r0-r1}          ;push x and n to stack
            SUB sp, sp, #4              ;reserve above parameters for the return value
            BL POWER                    ;call function power

            LDR r1, [fp, #-16]          ;get return value from above recursive call to
POWER
            MUL r1, r0, r1              ;multiply x * previous return value to r1
            B RETURN                    ;jump to RETURN

EVEN        ASR r1, #1                  ;divide n by 2
            STMFD sp!, {r0-r1}          ;push x and n to stack
            SUB sp, sp, #4              ;reserve above parameters for the return value
            BL POWER                    ;call power function
            LDR r0, [fp, #-16]          ;get returned value from above recursive call to
POWER
            STR r0, [fp, #-4]           ;store returned value in local variable y on the stack
            MUL r1, r0, r0              ;multiply y*y into r1

RETURN      STR r1, [fp, #16]           ;store r1 in return value address on stack
            MOV sp, fp                  ;move the stack pointer to the current frame
pointer
            LDMFD sp!, {r0-r1, fp, pc}  ;restore modified registers along with fp

            SPACE 666                   ;define stack space
STACK       DCD 0x00                    ;location of fdstack in memory
RESULT      DCD 0x00                    ;store final result

            END
```

PART TWO: STACK FRAME

Local variable y	← sp
R0	← fp
R1	
Fp	
Lr	
return value	
Parameter x	
Parameter n	

PART THREE: NUMBER OF FRAMES TO CALCULATE N POWER OF X

N=0	**→**	**1 frame**
N=1	**→**	**2 frames**
N=2	**→**	**3 frames**
N=3	**→**	**4 frames**
N=4	**→**	**4 frames**
N=5	**→**	**5 frames**
N=6	**→**	**5 frames**
N=7	**→**	**6 frames**
N=8	**→**	**5 frames**
N=9	**→**	**6 frames**
N=10	**→**	**6 frames**
N=11	**→**	**7 frames**
N=12	**→**	**6 frames**