Report 3

Stack Frame Used (FD)

sp
y
r3
r2
r1
r0
fp
lr
result
n
x

Number of stack frames required to calculate X^n

N	Stack frames used
0	1
1	2
2	3
3	4
4	4
5	5
6	5
7	6
8	5
9	6
10	6
11	7
12	6

Source code: power.s

```
AREA Power, CODE, READONLY

n               EQU   2
x               EQU   3
                ENTRY

main            ADR    sp, stack            ; define stack
                MOV r1, #x                  ; put first paramter n into register
                STR    r1, [sp, #-4]!       ; push n onto the stack
                MOV r0, #n                  ; put paramter x into register
                STR r0, [sp, #-4]!          ; push x onto the stack
                SUB    sp, sp, #8           ; make space on stack for result

                BL      powerFunc           ; call power function

                LDR r0, [sp], #4            ; pop result from stack and store in r0
                ADD sp, sp,#8               ; remove n and x from stack
                ADR r2, result              ; get address of result
                STR    r0, [r1]             ; store result in result variable

Loop            B Loop                      ; end here

powerFunc       STMFD sp!, {r0,r1,r2,r3,fp,lr}  ; push and save general registers
                MOV fp, sp                  ; set fp to current sp
                SUB    sp, sp, #4           ; making space for local variable y
                LDR r1, [fp, #36]           ; getting param x
                LDR r0, [fp, #32]           ; getting param n

                CMP r0, #0                  ; compare n to 0
                MOVEQ r0, #1                ; if equal then add 1
                STREQ r0, [fp, #28]         ; store result in result variable
                BEQ    return               ; jump to return function

                AND r2, r0, #1              ; and n with 1
                CMP r2, #1          ; compare result to one to see if odd or even

                                            ; if odd
                SUBEQ r0, #1                ; subtract one from n
                STREQ r0,[fp,#32]           ; update parameter
                SUBEQ sp, sp, #4            ; reserve space for return value
                BLEQ  powerFunc             ; jump to powerFunc
                LDREQ r1, [sp], #4          ; load result in r1
                ADDEQ sp, sp, #4            ; remove paramter from stack
                MULEQ r2, r0, r1            ; multiply frame result with x
                STREQ r2, [fp, #28]         ; return result to the stack
                BL return                   ;jump to return

                                            ; if even
                MOVNE r0, r0, LSR #1        ; divide n by two
                STRNE r0, [sp,#-4]          ; updating paramter
```

```
                    SUBNE sp, sp, #4            ; reserve space for return value
                    BLNE  powerFunc             ; jump to top of powerFunc
                    LDRNE r1, [sp], #4          ; load result in r1
                    ADDNE sp, sp, #4            ; remove parameter from stack
                    LDRNE r2, [sp, #4]          ; getting parameter y
                    MULNE r3, r2, r2            ; multiply y by y
                    STRNE r3, [fp, #28]         ; return result to the stack

return              MOV sp,fp                   ;collapse all activation frames
                    LDMFD sp!, {r0, r1,r2,r3, fp, pc};reload registers and return to caller

;-------------------------------------------------------------------------------
                    AREA Power, DATA, READWRITE

result      DCD     0x00                        ; space allocation for final result
                    SPACE 0xF8                  ; stack space
stack       DCD     0x00                        ; initial stack position

                    END
```