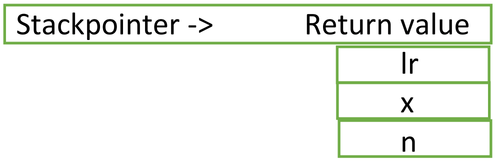
```
                             AREA power,CODE,READWRITE
                             ENTRY

MainProgram        ADR r0,result                          ;load address of result
                   MOV r1,2                               ;x
                   MOV r2,3                               ;n

                   STMDB sp!,{r1,r2}            ;store x and n into stack

                   BL PowerFunction             ;branch to power function

                   LDR r5,[sp],#4                         ;load the result from stack
                   STR r5,[r0],#1                         ;store result into result
Stop               B Stop                                          ;Stop loop


PowerFunction      STMDB sp!,{lr}               ;store link address in stack
                   SUB sp,sp,#4                 ;create space in stack for result
                   LDR r3,[sp,#12]              ;load n from stack

CheckZero          CMP r3,#0                                       ;if n is zero
                   MOVEQ r5,#1                                               ;then the result is 1
                   BEQ Return                                                ;branch to Return

CheckOdd           TST r3,#1                                       ;test for odd
                   BNE Oddn                                        ;if odd branch odd
                   B Evenn                                                   ;if even branch even

Oddn               LDR r4,[sp,#8]                                  ;load x into r4
                   SUB r3,r3,#1                                    ;subtract 1 from n
                   STMDB sp!,{r4,r3}                      ;store x and n to stack
                   BL PowerFunction                       ;branch to PowerFunction (recursive
call)
                   LDR r5,[sp],#4                                  ;load the result from stack
                   LDMIA sp!,{r4,r3}                      ;load the value of x and n from stack
                   MUL r5,r4,r5                                    ;multiply the result by x
                   B Return                                        ;branch to Return

Evenn              ASR r3,#1                                       ;if we have an even n then
halve n
                   LDR r4,[sp,#8]                                  ;load x into r4
                   STMDB sp!,{r4,r3}                      ;store x and n to stack
                   BL PowerFunction                       ;branch to PowerFunction (recursive
call)
                   LDR r5,[sp],#4                                  ;load the result from stack
                   LDMIA sp!,{r4,r3}                      ;load the values of x and n from stack
                   MOV r7,r5                                       ;create register to hold
result (y)
                   MUL r5,r7,r5                                    ;square the result (y)
                   B Return                                        ;branch to Return

Return             ADD sp,sp,#4                                    ;collapse frame
                   MOV pc,lr                                       ;Return to previous call


AREA power,DATA,READWRITE

x                                     DCB 2                                  ;x
n                                     DCB 3                                  ;n
result                       space 0xFFF                           ;space for result

END
```

Sketch:



Stackpointer -> Return value

lr

x

n

n-value	Number of stack frames to calculate x^n
0	1
1	2 (one more than 0n)
2	3 (one more than 1n because 2 is twice 1)
3	4 (one more than 2n)
4	4 (one more than 2n because 4 is twice 2)
5	5 (one more than 4n)
6	5 (one more than 3n because 6 is twice 3)
7	6 (one more than 6n)
8	5 (one more than 4n because 8 is twice 4)
9	6 (one more than 8n)
10	6 (one more than 5n because 10 is twice 5)
11	7 (one more than 10n)
12	6 (one more than 6n because 2 is twice 6)