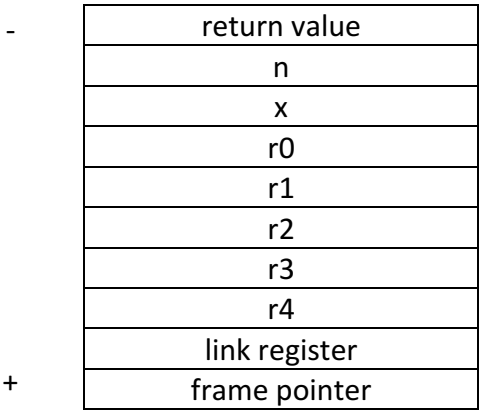
```
                AREA    power, CODE, READONLY
                ENTRY

                ;constants
x               EQU             2
n               EQU     5

MAIN    MOV             r0, #n          ;loads value of n to r0
LDR from memory
                MOV     r1, #x          ;load value of x to r1
LDR from memory
                ADR             sp, Stack               ;Points sp to
stack location
                STMFD           sp!, {r0,r1}    ;push x and n onto
stack
                SUB             sp, sp, #4              ;allocate
space for return call
                BL              PWR                     ;link to
function
                LDR     r4, [sp]        ;final value
                STR     result, r4              ;store it in result

ILOOP  B                ILOOP

PWR             SUB             sp, sp, #4              ;Create Stack Frame
                STR             fp, [sp]                ;push frame
pointer onto stack
                MOV     fp, sp          ;frame pointer points
to the base
                STMFD   sp!, {r0-r4,lr}         ;pushes value of r0 and link
register onto stack

                ; int n;
                LDR     r0, [fp,#8]
                ; int x;
                LDR     r2, [fp,#12]

                CMP             r0, #0          ;checks if base
case 0 is reached
                MOVEQ           r1, #1          ;If reached,
move 1 into r1 (result reg)
                SUB     sp, sp, #12
                BEQ             return          ;if base, pop
```

```
                    ;if (n & 1)
                    ANDS        r3, r0, #1              ;checks if odd (last bit
is 1)
                    BEQ             EVEN                ;branch to
even
                    SUB             r0, r0, #1          ;decerement n
by 1
                    STMFD           sp!, {r2,r0}        ;push x and n onto
stack {x,n}
                    SUB             sp, sp, #4          ;allocate
space for return call
                    BL              PWR
                    LDR         r4, [sp]            ;load value
                    MUL         r1, r4, r0              ;r1 = x*power(x,n-1)
                    B               return              ;branch to
return

EVEN        ASR         r0, r0, #1                  ;if not base case, n >> 1 (half)

                    B               PWR                 ;call power(x,
n >> 1)
                    MUL         r2, r1, r1              ;take return value y
and square it
                    STMFD           sp!, {r2,r0}        ;push x and n onto
stack {x,n}
                    SUB             sp, sp, #4          ;allocate
space for return call
                    BL              PWR                 ;recursive call
                    LDR         r4, [sp]            ;load value
                    MUL         r1, r4, r4              ;y = return value,
store y*y in r1


return      STR             r1, [fp, #4]        ;
                    ADD         sp, sp, #12
                    LDMFD       sp!, {r0-r4, lr}
                    MOV         sp, fp                  ;restore stack pointer
                    LDR         fp, [sp]            ;restore old frame pointer
from stack
                    ADD         sp, sp, #4              ;move stack pointer
down by 4 bytes
                    MOV         pc, lr                  ;return by loading lr
into pc
```

```
                AREA    power, DATA, READWRITE
                ENTRY

Stack           DCD             0x00
result          DCD         0x1000


                END
```

Stack Frame



How many stack frames needed to calculate x^n?

N	STACK FRAMES
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