AREA question1, CODE, READONLY

ENTRY

X EQU 4 ;define X

N EQU 3 ;define N

main ADR sp, STACK ;define the stack

MOV r0, #X ;pass the value of x to power function store in r0

MOV r1, #N ;pass the value of n to power function store in r1

STMFD sp!, {r0-r1} ;pushing x and n into stack by store full decending

SUB sp, sp, #4 ;decrease the stack pointer by 4 bytes

BL power ;branch to function power

LDR r2, [sp], #12 ;when subroutine power returns, return value is at top of stack, load into r2

ADR r3, result ;get address of result

STR r2, [r3] ;store return value of power in result

loop B loop ;the program done

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] ;get the parameter n via stack

CMP r1, #0 ;check if n is 0

MOVEQ r1, #1 ;if n is 0, return 1(because x^0 is 1)

BEQ RETURN ;then go to RETURN

LDR r0, [fp, #20] ;if n is not 0,load the x

TST r1,#1 ;checking the last bit of r1 if it is 1

BNE ODD ;if n is odd, go to ODD

B EVEN ;else n is even, so go to EVEN

ODD SUB r1, r1, #1 ;decrement n by 1

STMFD sp!, {r0-r1} ;pass x and n via stack

SUB sp, sp, #4 ;leave an open spot above parameters for the return value

BL power ;call power function

LDR r1, [fp, #-16] ;get the value after power function

MUL r1, r0, r1 ;multiply x and previous x value, then store in to r1

B return ;go to RETURN

EVEN ASR r1, #1 ;arithmetic right shift one bit equal to divide by 2

STMFD sp!, {r0-r1} ;pass x and n via stack

SUB sp, sp, #4 ;leave an open spot above parameters for the return value

BL power ;call power function

LDR r0, [fp, #-16] ;get the value after power function

STR r0, [fp, #-4] ;store the value of y into stack

MUL r1, r0, r0 ;multiply y\*y and then store it into r1

return STR r1, [fp, #16] ;store r1 to stack

MOV sp, fp ;move the stack pointer to frame pointer

LDMFD sp!, {r0-r1, fp, pc} ;restore modified registers along with fp, and move lr into pc

SPACE 512 ;we may get a many numbers in stack, be sure we have enough space store tham

result DCD 0x00 ;to store final result of x^n

STACK DCD 0x00 ;create a empty stack

END

