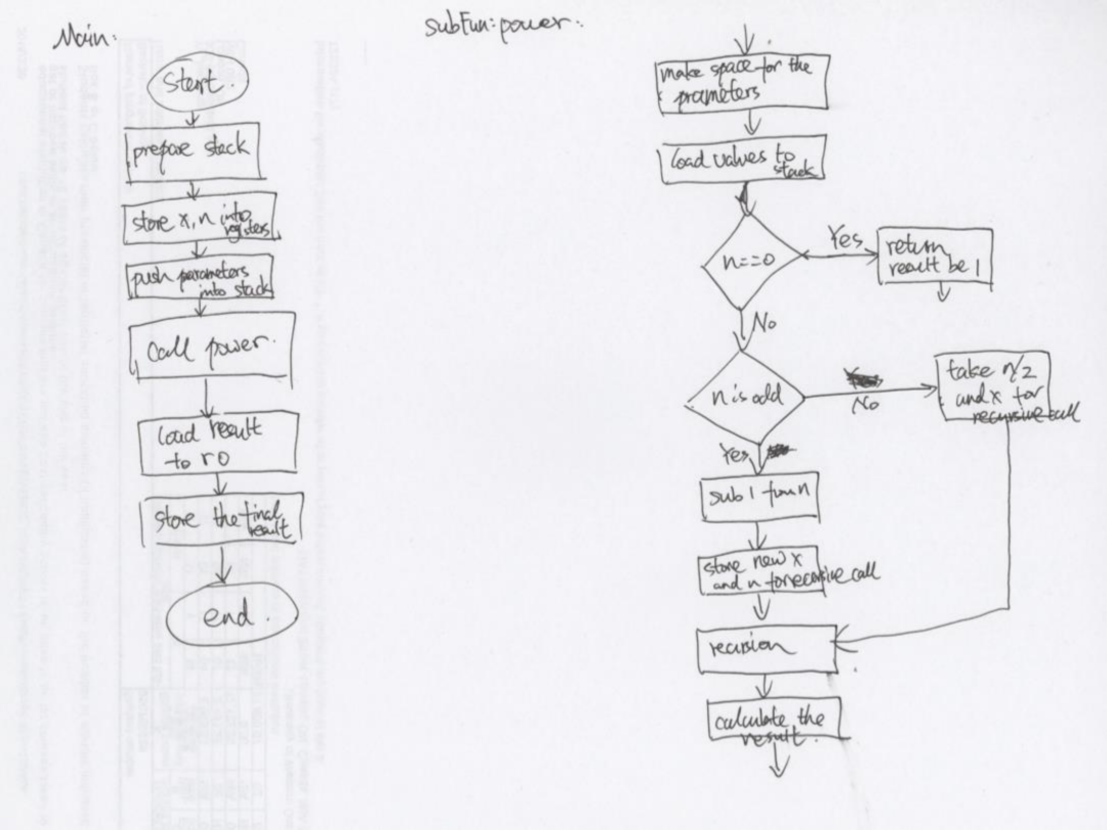
```
        AREA Assignment5, CODE, READONLY

        ENTRY
x       EQU 2
n       EQU 2
Main    ADR sp, stack                 ;define the stack
        MOV r0, #x                         ;prepare the parameter, store x in R0
        MOV r1, #n                         ;prepare the parameter, store n in R1
        STMFD sp!, {r0, r1}           ;push the parameters on the stack
        SUB sp,sp,#4                  ;reserve a place in the stack for the return value
        BL power                           ;call the power subroutine
        LDR r0, [sp], #4              ;load the result in r0 and pop it from the stack
        ADD sp,sp,#8                  ;also remove the parameter from the stack
        ADR r1, result                     ;get the address of the result variable
        STR R0, [R1]                  ;store the final result in the result variable
Loop    B Loop                             ;infinite loop

        AREA Assignment5, CODE, READONLY
power   STMFD sp!,{r0-r2,fp,lr}   ;push general registers, as well as fp and lr
        MOV fp, sp                         ;set the fp for this call
        SUB sp, sp, #12               ;create space for the x and y local variables
        LDR r0, [fp, #0x1C]           ;get the parameter from the stack
        LDR r1, [fp, #0x18]           ;get the parameter from the stack
        CMP r1, #0                         ;test the base condition for recurssion
        MOVEQ r0, #1                       ;if n == 0, store 1 to the register
        BEQ Return                         ;jump to the return part
        LSRS r2, r1, #1               ;test whether n is even or odd
        BCC else                      ;if n is even, do the else branch
        SUB r1,r1,#1                  ;if n is odd, subtract 1 from n
        STMFD fp, {r0, r1}            ;store parameter x and n for recurssive call
        BL power                           ;call subroutine
        LDR r1, [fp, #-12]            ;load the returned value to R1
        MUL r2, r0, r1                     ;multiply the returned value with x
        B Return                           ;jump to the return part
else    STMFD fp, {r0, r2}            ;store parameter x and n for recurssive call
        BL power                           ;call subroutine
        LDR r1, [fp, #-12]            ;load the returned value to R1
        MUL r2, r1, r1                     ;square the returned value
Return  STR r2, [fp, #20]             ;store the returned value in the stack
        ADD sp, #12                   ;remove also the parameter from the stack
        LDMFD sp!,{r0-r2,fp,pc}       ;load all registers and return to the caller

        AREA Assignment5, DATA, READWRITE
```

```
result	DCD 0x00				;the final reault
	SPACE	0xF0				;declare the space for stack
stack	DCD 0x00				;initial stack position
	END
```


Main:
Start.
prepare stack
store x, n into registers
push parameters into stack
call power.
load result to r0
store the final result
end.
subFun: power:
make space for the prameters
load values to stack
n==0
Yes
return result be 1
No
n is odd
No
take n/2 and x for recursive call
Yes
sub 1 from n
store new x and n for recursive call
recursion
calculate the result.