Meiling Zhang
250907723
mzhan387
2208 Assignment5

```
           AREA power, CODE, READWRITE
           ENTRY
Main        MOV r0,#x          ;store value of x in r0
           MOV r1,#n          ;store value of n in r1
           LDR sp, =stack     ;load the stack pointer into stack
           LDR fp, =stack     ;load the frame pointer into stack
           STMIB sp!, {r0, r1, r2}    ;store multiple register into stack using
the order of increment before
           BL Function        ;recall function
           LDR r2, [fp, #12] ;load the address of result value in r2
           STR r2, result     ;store the value of result in r2
Loop  B Loop       ;end the loop

Function
            STMIB sp!,{r0,r1,fp,lr}    ;store function needed register into
stack
           LDMIB fp, {r0,r1}            ;r0=x, r1=n
           MOV fp, sp     ; store stack pointer into frame pointer

           CMP r1,#0     ;check if r1 equal to zero
           MOVEQ r1,#1   ;if equal, set r1 to one
           STREQ r1,[fp,#-16]  ; if equal, store the address of frame pointer
to r1
           MOVEQ sp, fp    ;if equal, store the frame pointer into stack
pointer
           LDMDAEQ sp,{r0,r1,fp,pc}   ;if equal, load data

           TST r1, #1  ; test if r1 euqal to 1
           BEQ Even    ;if equal, jump to even, else to odd
Odd        SUB r1, #1   ;subtract 1 from r1
           STMIB sp!, {r0, r1, r2}    ;store multiple register into stack using
the order of increment before
           BL Function      ;recall function
           LDR r1, [fp,#12]    ;load the address of result value in r1
           MUL r0,r1,r0        ;multiply r0 and r1 and store in r0
           STR r0,[fp,#-16]    ;store the address of result value in r0
           MOV sp, fp          ;sotre frame pointer into stack pointer
           LDMDA sp,{r0,r1,fp,pc}      ;load data

Even      LSR r1,#1
          STMIB sp!, {r0, r1, r2}      ;store multiple register into stack
using the order of increment before
```

```
           BL Function       ;recall function
           LDR r1, [fp,#12]     ;load the address of result value in r1
           MUL r0,r1,r1      ;squre r1 and store the result in r0
           STR r0,[fp,#-16]    ;store the address of result value in r0
           MOV sp, fp      ; sotre frame pointer into stack pointer
           LDMDA sp,{r0,r1,fp,pc}     ; ;load data

n          EQU 12         ;assign the value of n
x          EQU 2          ;assign the value of x
result     SPACE 4        ;create space for result
stack      DCD 0x00       ;leave space for stack

           END
```

Stack frame of function call #1	Returned value from function call #2	i + 40
	Function Call #1 n	i + 36
	Function Call #1 X	i + 32
Stored registers & return address	LR	i + 28
	FP	i + 24
	R1	i + 20
	R0	i + 16
Main method	Returned value from function call #1	i + 12
	Original N	i + 8
	Original X	i + 4
	0x00	i + 0

STACK

Stack growth: Ascending
Class: Full
Stack suffix: FA
Load suffix: DA (decrement after)
Store suffix: IB (increment before)