# Solutions to Tutorial 3

## 3.1 Stack Manipulation Operations

(1) `SP = 0xFEC`
(2) `STR R1,[SP,#-4]! Or STMFD SP!,{R1}`
(3) `LDMFD SP!,{R6-R9}`
(4) `LDR R0,[SP,#8]`

## 3.2 Modular Programming – Subroutine Call and Parameter Passing

(1) **`NumX`** and **`NumY`** are passed by value. Values of memory variables are pushed to the stack. **`Ans`** is passed using reference. Address of the memory variable pushed onto the stack.

(2) After (b1) **`PC=0x014 SP=0xFFFFFFF8`**

After (c1) **`PC=0x040 SP=0xFFFFFFF0`**

After (s6) **`PC=0x028 SP=0xFFFFFFF0`**

(3) **`R0=Value of R3 (can be 0)`**

(4) **`ADD SP,SP,#12`**

(5) Replace **`Bl MySub`** with **`MOV LR,PC`** followed by **`B MySub`**.

(6) Suggested solutions:

```
MySub   STMFD SP!,{R0-R3}     ; (s1) Save registers R0,R1,R2,R3
        LDR   R0,[SP,#24]     ; (s2) Retrieve NumX from stack
        LDR   R1,[SP,#20]     ; (s3) Retrieve NumY from stack
        MOV   R2,#0           ; Complete the segment of code to compute the
Loop    ADD   R2,R2,R1        ; value of NumX*NumY using successive addition
        SUBS  R0,R0,#1        ; decrement NumY till reach zero
        BNE   Loop            ; loop back till NumX added NumY times
        LDR   R3,[SP,#16]     ; (s4a) Move the result directly to …
        STR   R2,[R3]         ; (s4b) … the memory variable Ans
        LDMFD SP!,{R0-R3}     ; (s5) Restore saved registers
        MOV   PC,LR                   ; (s6) Return to calling program
```