Stack Models

```
0x418
Inc. memory location

0x400
Old SP

0x3e8
0x3ec
```

```
Stack grows
```

```
SP
```

```
r5
r4
r3
r1
r0
```

```
0x3e9
```

```
Full 4 byte
Descending
```

```
Reads 4 bytes from the top of the stack
```

```
Empty
```

```
Stack grows
```

```
SP
```

```
r5
r4
r3
r1
r0
```

```
Full
```

```
Empty
```

```
Stack grows
```

```
SP
```

```
r5
r4
r3
r1
r0
```

```
Old SP
```

```
0x3e9
```

```
(1) (2) (3) (4)
```

```
UCSD
```
ARM Procedure call standard

The AAPCS specifies a

- Full descending stack
- Stack is 8 byte aligned
The following ARM statements are translations of the given C code, variables i and j should be maintained on the stack.

Q: Is the state of memory the same after each code is executed?

\[ \text{int } i = 10, j = 20; \]
\[ i = i + j; \]

A. Yes
B. No
New instruction STM

STMDB sp!, {r0 – r1}

Multiple
Decrement before
Overwrite SP
Starting location

foo:
MOV r0, #10
MOV r1, #20
ADD sp, sp, #8
STR r0, [sp]
STR r1, [sp, #4]
ADD r0, r0, r1
BX lr

foo:
MOV r0, #10
MOV r1, #20
STMDB sp!, {r0, r1}
ADD r0, r0, r1
BX lr

push \{r0, r1\}
C local variables

Is the given C and ARM code equivalent?

A. Yes
B. No

foo:
MOV r0, #10
MOV r1, #20
STMDB sp!, {r0, r1}
ADD r0, r0, r1
BX lr

int foo()
{
int i=10, j=20;
return i+j;
}

Fix to return the stack to its original form
Recursion Example

int fact (int n)
{
    if (n <= 1)
        return 1;
    else
        return n * fact(n - 1);
}

\[ n! = n \times (n-1)! \quad n > 1 \]

- Base case: \( n \leq 1 \)
- Recursive step: \( n > 1 \)

\[ \begin{align*}
    \text{fact(1)} & \rightarrow \text{return 1} \\
    \text{fact(2)} & \rightarrow 2 \times \text{fact(1)} \\
    \text{fact(3)} & \rightarrow 3 \times \text{fact(2)}
\end{align*} \]
Recursion in ARM

fact:
CMP r0, #1
BLE ret_one
MOV r1, r0
SUB r0, r0, #1
BL fact
MUL r0, r0, r1
B end

ret_one: MOV r0, #1
end:
BX lr

int fact (int n)
{
    if (n <= 1)
        return 1;
    else
        return n * fact(n - 1);
}

What is the value returned by fact(1)?
A. One
B. Two
C. Three
D. Six
E. None of the above

main: mov r0, #1
BL fact
mov r4, r0
Recursion in ARM

fact:

CMP r0, #1
BLE ret_one
MOV r1, r0
SUB r0, r0, #1
BL fact
MUL r0, r0, r1
BX lr

ret_one: MOV r0, #1
end:

What is the value returned by fact(2)?

A. One  
B. Two  
C. Four  
D. Six  
E. None of the above

int fact (int n)
{
    if (n <= 1)
        return 1;
    else
        return n * fact(n - 1);
}

main: mov r0, #2
BL fact
mov r4, r0
Recursion in ARM

**fact:** push `{lr}

CMP r0, #1
BLE ret_one
MOV r1, r0
SUB r0, r0, #1
BL fact
MUL r0, r0, r1
B end

**ret_one:** MOV r0, #1
end: pop `{lr}
BX lr

```c
int fact (int n)
{
    if (n <= 1)
        return 1;
    else
        return n * fact(n - 1);
}
```

What is the value returned by fact(3)?
A. One
B. Two
C. Four
D. Six
E. None of the above
Recursion in ARM

fact:  push {r1, lr}
       CMP r0, #1
       BLE ret_one
       MOV r1, r0
       SUB r0, r0, #1
       BL fact
       MUL r0, r0, r1
       B end

ret_one: MOV r0, #1
end:     pop {r1, lr}
        BX lr

int fact (int n)
{
    if (n <= 1)
        return 1;
    else
        return n * fact(n - 1);
}

What is the value returned by fact(3)?
A. One
B. Two
C. Four
D. Six
E. None of the above
int read_and_sum() {
    int l1, l2;
    scanf("%d%d", &l1, &l2);
    return l1 + l2;
}

read_int:
    push {r4-r11, ip, lr}
    sub sp, sp, #8  @ reserve 8 bytes for the 2 local integers

    ldr r0, =scan_ints  @ format string
    mov r1, sp        @ address of l1
    add, r2, sp, #4   @ address of l2
    bl scanf

    ldr r0, [sp]      @ read l1
    ldr r1, [sp, #4]  @ read l2
    add r0, r0, r1

    add sp, sp, #8    @ de-allocate 8 bytes, restore sp to
                      @ value after push

    pop {r4-r11, ip, lr}
    bx lr