**Fill out this document, copy all items from Concepts 1 through 8, and the Comments/Questions, and paste into your created discussion thread.**

1. *C++*

* Function
  + Ex) Int mystery (int x, int y){

x = 5;

return x+y;

}

* + Output:
    - a = 10, b = 5
    - Mystery(a, b)
    - cout <<a <<endl;
  + Ex) Int mystery (int&x, int y){

x = 5;

return x+y;

}

* + Output:
    - a = 10, b = 5
    - Mystery(a, b)
    - cout <<a <<endl;

*Asm*

* output : . asciz “Sum of %d and %d is %d\n”

…

ldr r0, =output //Pass by references

mov r1, #1 //Pass by value

mov r2, #5 //Pass by value

add r3, r1, r2 //Pass by value = 6

bl printf

.data

X: .word 123

Out : .asciz “x is at address %d, and contains %d\n”

…

Ldr r0, =out

Ldr r1, =x

Ldr r2, [r1]

Bl print f

Affected registers: r0,r1,r2,r3

Code results: Passes values and adds them = 6

1. Add range function

* add\_range: //Passed by value

//Doesn’t require push {lr} since lr doesn’t change

//First param r0, second r1

mov r2, #0 //Sum

mov r3, r0 //r3 = lower\_limit

loop:

cmp r3, r1

bgt done

add r2, r3

add r3, #1

bal loop

done:

mov r0, r2

mov pc, lr

* add range 2: //add\_range2: pass lower and upper limit by reference

push {r4} //Need another register for values

mov r2, #0

ldr r3, [r0]

ldr r4, [r1]

loop2:

cmp r0, r1

bgt done\_2

add r2, r0,

add r0, #1

bal loop2

done:

mov r0,r2

mov pc,lr

Affected registers: r0, r1,r2

Code results: Passes values to functions either by reference or value

1. Recursion

*C++*

* Int fact(int n){

If(n==0) return 1;

Else

Return n\* fact(n-1);

}

*Asm*

* Factorial:

push {lr}

cmp r0, #0 //Compares registry and value

beq done //If equal branch to done

push {r0} //Preserves r0 call

sub r0, #1 //Fact(r0 -1)

bl fact

pop {r1}

mul r0, r1

pop {pc}

Affected registers:

Code results:

4.

Affected registers:

Code results:

5.

Affected registers:

Code results:

6.

Affected registers:

Code results:

7.

Affected registers:

Code results:

8.

Affected registers:

Code results:

Comments/Questions?