Michael Palumbo
CS-281-B
You can find the mips and c code within the folder named palumbomichael.asm and palumbomichael.c respectively.

Q: What do fun1 and fun2 do?
What we want to do is run through fun1 and fun2, but when we hit fun2 instead of going back to its expected return address (if fun2 was called on line 8, it returns back to line 8) we return to fun1's return address.

This can be achieved quite simply but also sneakily. The return address of functions are located 1 place to the left of the arguments location in memory. So if we would want to grab the return address of fun1, we simply ask where the argument of fun1 is, in our case x, by typing '&x' and then minus it by 1. And since we want to remember the return address once we leave fun1, we save it to a variable, in our case global_ra.
We then encounter fun2, where global_ra now stores the return address of fun1. So to set fun2 address to fun1's we say *(&x - 1) = global_ra. Then when we hit the return statement, the code will look at &x-1 and go to fun1's return address.

The Stack:
Main Before Fun1 or Fun2 is called

before addiu $sp, $sp, -40		after addiu $sp, $sp, -40		
counting words	counting bytes	counting words	counting bytes	
-10	original sp-40	new sp	new sp +0	
-9		1		
-8		2		
-7		3		
-6		4		
-5		5		
-4		6		
-3		7		
-2		8	32	fp
-1		9	36	ra
	original sp	10	40	argc
		11	44	argv

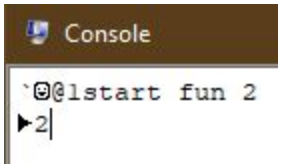
Before Fun1 is returned

before addiu $sp, $sp, -32			after addiu $sp, $sp, -32		
counting words	counting bytes		counting words	counting bytes	
new sp	new sp +0		new sp	new sp +0	
1			1		
2			2		
3			3		
4			4		
5			5		
6			6		
7			7		
8	32	fp	8	32	fp
9	36	ra	9	36	ra
10	40	argc	10	40	argc
11	44	argv	11	44 (4)	argv
			12	8	
			13	12	
			14	16	
			15	20	
			16	24	fp
			17	28	ra
			18	32	x

Before Fun2 is called (note we pop off everything that happened in fun 1)

before addiu $sp, $sp, -32			after addiu $sp, $sp, -32		
counting words	counting bytes		counting words	counting bytes	
new sp	new sp +0		new sp	new sp +0	
1			1		
2			2		
3			3		
4			4		
5			5		
6	24	v (this happened before fun2 and after fun1)	6		v
7			7		
8	32	fp	8	32	fp
9	36	ra	9	36	ra
10	40	argc	10	40	argc
11	44	argv	11	44 (4)	argv
			12	8	
			13	12	
			14	16	
			15	20	
			16	24	fp
			17	28	ra
			18	32	x

Console output:



(i honestly don't know where the stuff before `start fun 2` came from)

Also incase it asked for something like compiler output