Fun with Stacks

The stack is a "last in, first out" structure. It is important to initialize the stack pointer so that your stack is large enough to contain all the bytes that you will push onto it. The stack pointer is initialized to 0x07, so the first value pushed onto the stack is stored at 0x08, which is the value of R0 in register bank 1. If more than one register bank is to be used, the stack pointer should be moved at the beginning of the program. Choose a location in the data memory that is not being used. We will use location 0x30 as the first stack data value, so we will initialize the stack to 2Fh.

Write a program that initializes the stack pointer, and then pushes the following 4 memory locations on the stack, and then pops them back out so that the data is re-ordered. The following "before" and "after" example illustrates how it should work.

Before		**After**	
Address:	Data	Address:	Data
70	34	70	89
71	f2	71	66
72	66	72	f2
73	89	73	34

Since you do not know what data is initially in RAM, add some instructions to initialize these memory locations so that you can tell if your program works. (If the locations all held the same data value, then you would not be able to see the changes). Name your program **stacks.asm**