Purpose: Write a MIPS function that uses a stack.

Details: You will complete a C program and complete two MIPS programs (all have been begun for you and are in the shared repository in a folder named lab6).

The programs must be fully commented; comments should include your name, the lab number, the problem number, the lab date, the Honor Code pledge, and a description of what the program does. The MIPS code must be documented for each part of the program. Imitate the commenting style of the sample programs.

1. Write a rotate function in C. In the class shared repository, in folder lab6, get a copy of the program “rotate.c.” This program is missing the body of function rotl. Function rotl is supposed to rotate the contents of an array one place to the left (see sample output below).

Finish writing the rotl function. You should not change any of the code in main or in aprint; you should not call aprint (that’s already been done for you). Your program’s output should look just like this:

```
$ ./a.out
Unrotated array: 10 20 30 40
Rotated array: 20 30 40 10
Unrotated array: 3
Rotated array: 3
Unrotated array: 11 12 13 14 15 16 17 18 19
Rotated array: 12 13 14 15 16 17 18 19 11
```

2. Write a MIPS program that rotates an array. In the shared class repository in folder lab6, get a copy of the program “rotate1.asm”. Complete it so that it rotates the contents of the array stored at location a1. You do not need to write a MIPS function; you do not need to use jal or jr or the stack or anything like that. (You’ll do that in part 3.) You will need a loop that performs the same rotation described in part 1. Try to make your loop general—only use the constant “4” when setting the loop limit; the rest of your code should be general enough to work with any size array.

The output of your MIPS program should look like this:
Unrotated array: 10 20 30 40
Rotated array: 20 30 40 10

-- program is finished running --

3. Convert your loop into a MIPS function. In the shared class repository in folder lab6, get a copy of the program “rotate2.asm”. Complete the function rotl at the bottom of the program so that it will rotate any integer array whose address is provided in register $a0 and whose size is provided in register $a1. **IMPORTANT:** YOU ARE REQUIRED TO USE THE “s” REGISTERS FOR THINGS LIKE LOOP COUNTER AND OTHER IMPORTANT PROGRAM VARIABLES. YOU ARE REQUIRED TO PRESERVE THE s REGISTERS ON THE STACK AT THE BEGINNING OF THE FUNCTION AND TO RESTORE THEM FROM THE STACK BEFORE RETURNING. Imitate the form of function aprint in the file provided.

You should not change any of the other code; you should not call aprint (that’s already been done for you). Your program’s output should look just like this:

Unrotated array: 10 20 30 40
Rotated array: 20 30 40 10
Unrotated array: 3
Rotated array: 3
Unrotated array: 11 12 13 14 15 16 17 18 19
Rotated array: 12 13 14 15 16 17 18 19 11

-- program is finished running --

Submit the fully commented C program and MIPS programs by 8 a.m. on October 19.