Homework 1

.globl main

.data

x: .word 0

str: .word str\_data

str\_data: .asciiz "Input a positive integer: "

strout1: .word strout1\_data

strout1\_data: .asciiz "The value of factorial "

strout2: .word str\_strout2\_data

strout2\_data: .asciiz "is "

.text

main:

# The C program refers only to str, not to str\_data

# So, this verion will use only str, and not str\_data, in the code.

la $t0, str # Load address of str (It's value will point to string)

lw $a0, 0($t0) # Given address in $t0, load its value into $a0

# Before, we saved $ra in $s0; That wouldn't work if the program

# were recursive. So, now, we demonstrate saving on stack

addi $sp, $sp, -4 # Extend stack; Use -4 because stack grows downwards

sw $ra, 0($sp) # Save $ra on stack

jal print\_mystring # Call print\_mystring(str)

lw $ra, 0($sp) # Restore $ra from stack

li $v0, 5 #Request an input number from the user

syscall

lw $t1, $v0

lw $a0, $v0 #Load the input number into $a0

jal factorial # Call factorial with the input number as argument

addi $sp,$sp, 4 # Contract stack to where it used to be

# $v0 is return value

sw $v0, x($zero) # $v0 is return value; save in variable x

jr $ra

print\_mystring:

# argument to print\_str syscall is already in $a0

li $v0, 4 # The print\_str system call is number 4 in the table.

syscall

li $v0, 0 # 0 means success; Pass return value in $v0

jr $ra

factorial:

beq $a0, 0,