Laboratory 6: Inlab

Date ____________________ Section ____________________

Name ____

Floating Point Arithmetic in MIPS (cont' d)

This inlab session will focus on doing floating-point arithmetic and on various conversions.

Step 1

During Lab #4 you created a program named *lab4.5.asm* which computes the factorial of an integer number. All operations were performed on integer numbers. You are to create a new program named *lab6.11.asm* which computes the factorial of an integer using floating point numbers:

- in 'main' prompt the user to enter an integer
- check if the number entered is negative: if it is negative, then print a message saying so and prompt the user again for a number
- call the procedure named 'FactorialSingle' whose parameter will be the number read from the user, converted to single precision floating-point; the procedure returns the factorial (a single precision floating-point) of that number
- use the MIPS convention for procedure call
- prints a message and the value returned by 'FactorialSingle'

Run *lab4.5.asm* and the new program to complete the following test plan. Use scientific notation for the output printed by *lab6.11.asm* (i.e. 3.141592e0).

Test plan for *lab6.11.asm*

Input number	*lab4.5.asm* output	*lab6.11.asm* output
0		
5		
10		
15		
20		
40		

Q 1:

Why some of the results printed by *lab4.5.asm* are negative?’

Q 2:

What are the maximum values of the input for which correct output is still printed?’

Integer	n =
Single precision floating-point	n =

Step 2

Using some other method, calculate the exact value of 20! On a UNIX platform you can use bc (the “arbitrary-precision arithmetic language”). If no software is available to you to do the job, then you will have to use the old-good pencil and paper method.

The exact value for 20!

`20!=`

Run again *lab6.11.asm* and write down the value printed for 20!

Value printed by *lab6.11.asm* **for 20!**

`20!=`

Q 3:

Why do the two values differ? ’

Q 4:

How many digits are exact in the result printed by *lab6.11.asm*?

exact_digits =

Q 5:

Based on the answer to the previous question, can you make a rough estimation of the number of digits that should be printed for a floating-point single precision number?

digits_to_print =

Step 3

In step 1 you have used the integer to floating point conversion (`cvt.s.w`). Let's now try a conversion from float to integer. Modify *lab6.11.asm* (save the new program as *lab6.12.asm*):

- after printing the value returned by 'FactorialSingle', convert that value to an integer and print it too.

Run the program and complete the next test plan. Write down the value printed for the factorial as is, don't use scientific notation this time.

Test plan for *lab6.12.asm*

Input number	Floating-point output	Integer Output
`0`		
`5`		
`10`		
`11`		
`12`		
`13`		
`14`		
`15`		

Highlight those cases where the integer output is a number different than the floating-point output.

Q 6:

Do you think the conversion operation yields a signed integer or an unsigned one? Explain.

Q 7:

As you can see the conversion instruction does not signal any error even if the conversion itself results in a wrong value for the result. What is, in your opinion, the reason the architecture does not specify the conversion instructions should report errors? '

Q 8:

What would be the sequence of instructions that would emulate a signaling conversion? Do this only for the float to integer conversion.