Mips Lab 2

1. Topic: Finding the biggest number.
   A data list is given in the data memory. Each data is sized in bytes. The list begins at the address located in the register $a0 and its length is given in register $a1. Assume that the data list is signed.
   Find the largest number and enter it into register $v0. Display the largest data on your computer screen.
   Remarks:
   1. Each program should be accompanied by a flowchart (and the algorithm of the solution if necessary).
   2. Comments in the body of the program are mandatory.
2. Write a program for the implementation of a simple calculator according to the following rules:
   A) The computer will display the following message: ENTER VALUE. In response the user enters a decimal value.
   The data will be input as an INTEGER.
   B) The computer displays a message on the screen: ENTER OP-CODE . The input will be in the CHAR format and is one of 4 options: *, - , + , @ (multiply, subtract, addition, end the program, respectively).
   C) If the user ends the program, the result of the calculation should be displayed with the accompanying message :
   The result is
   D) If a non-termination action code is requested, then:
   The computer will display the following message: ENTER VALUE In response, the user will enter a decimal value.
   The data will be input as INTEGER.
   Assume that the maximum addition and subtraction operations are 32 bit.

A problem may occur when multiplying two numbers. A result greater than 32 bits may be received. There are 2 cases which will indicate such an “overflow”:
A. If the multiplication result is positive, then if the HI register is different from 0, it means that the result is greater than 32 Bits.
B. If the multiplication result is negative then if the entire HI register is not all ones then there was an overflow.

If the multiplication result is greater than 32 bits, the calculator must be stopped and abnormalities declared. Print “ Overflow detected”

Reminder: An Ascii value of any character can be loaded into a register.
For example: If I want to load the ASCII value of a character, I could perform the following instruction: `li $2, '-'`

numbers to check

352321696

*

256

hi and lo are positive but is overflow

214747954

*

11

Low is negative but hi is positive

7

-

7

*

9 = 0