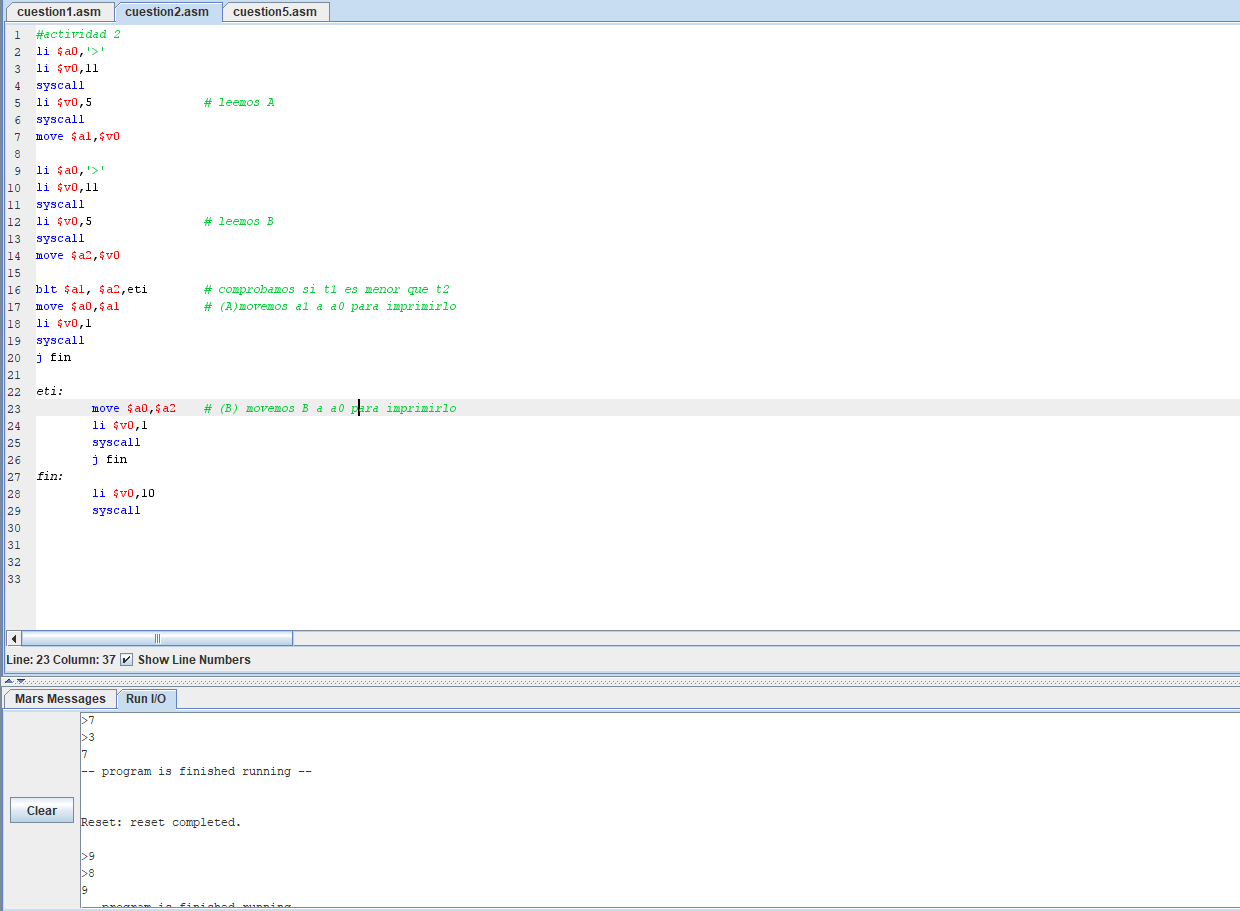
Qüestió 2

* Escriviu el programa que llig dos enters del teclat i escriu en la consola el més gran. El programa ha de tenir l’estructura següent:

Assembla i prova el programa amb diferents valors d’A i B



Lo que he hecho ha sido primero, pedir dos números con el syscall 5, después he utilizado blt para que salte a “eti” cuando el numero A sea menor que B. Dentro de eti movemos el valor de B a $a0 para poder imprimirlo por pantalla y después saltamos a la etiqueta para finalizar el programa.

En caso de que B sea menor que A no saltaríamos a eti y seguiríamos con la instrucción siguiente, que seria mover el valor de A a $a0 para imprimir por pantalla y finalizaríamos el programa.

Qüestió 3

 Observeu el programa assemblat que acabeu d’escriure corresponent a la qüestió 2 i ompli la plantilla de la figura 1 amb els camps en binari de la instrucció beq segons el format tipus I; escriviu primer la instrucció que codificareu.

Qüestió 4

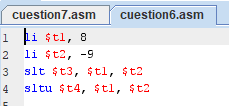
 Observeu de nou el programa assemblat que acabeu d’escriure corresponent a la qüestió 2 i ompliu la plantilla de la figura 2 amb els diferents camps en binari de la instrucció slt segons el format R. Escriu primer la instrucció a codificar.

Qüestió 5

 Escriviu un codi exemple per a esbrinar com tradueix l’assemblador les pseudoinstruccions següents: sgt, sge, ble.

Qüestió 6

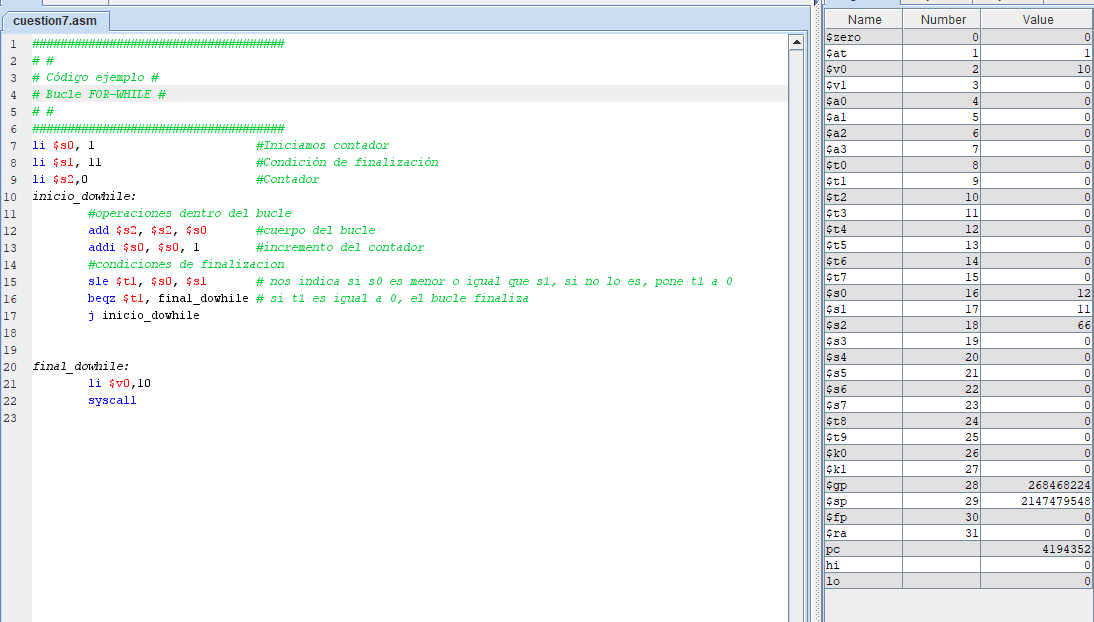
 Penseu en un cas en el qual slt i sltu donen resultats diferents per als mateixos continguts dels registres. Comproveu-ho.



Sltu al realizar la comprobación sin signo, nos dice que t2 es mayor que t1, todo lo contrario que slt que nos dice que el mayor es 51

Qüestió 7

 En exemples com l’anterior, en què sabem que almenys el bucle s’executa una vegada, és més eficient posar la condició d’eixida del bucle al final. Reescriviu el codi anterior amb la condició d’eixida al final del bucle (es tracta d’un bucle del tipus do-while).



Lo que he hecho aquí ha sido modificar el cogido del ejemplo anterior de tal manera que haga primero las operaciones y después que haga las comprobaciones. Por lo que pasaríamos de un for a un do-while.

Al inicio del bucle realiza la operación e incrementamos el contador, después comprueba si el contador y el limite del contador son iguales. En caso de que lo fueran el registro $t1 se pondría a 0, en caso contrario a 1.

Por ultimo se comprueba si $t1 es igual a 0, es decir si el contador ha llegado a su limite o no. Si no ha llegado al límite, saltará de nuevo a la etiqueta inicio\_dowhile y volverá a ejecutar lo mismo que antes. En caso de que hubiera llegado irá a la etiqueta final\_dowhile y finalizará el programa.

Qüestió 8

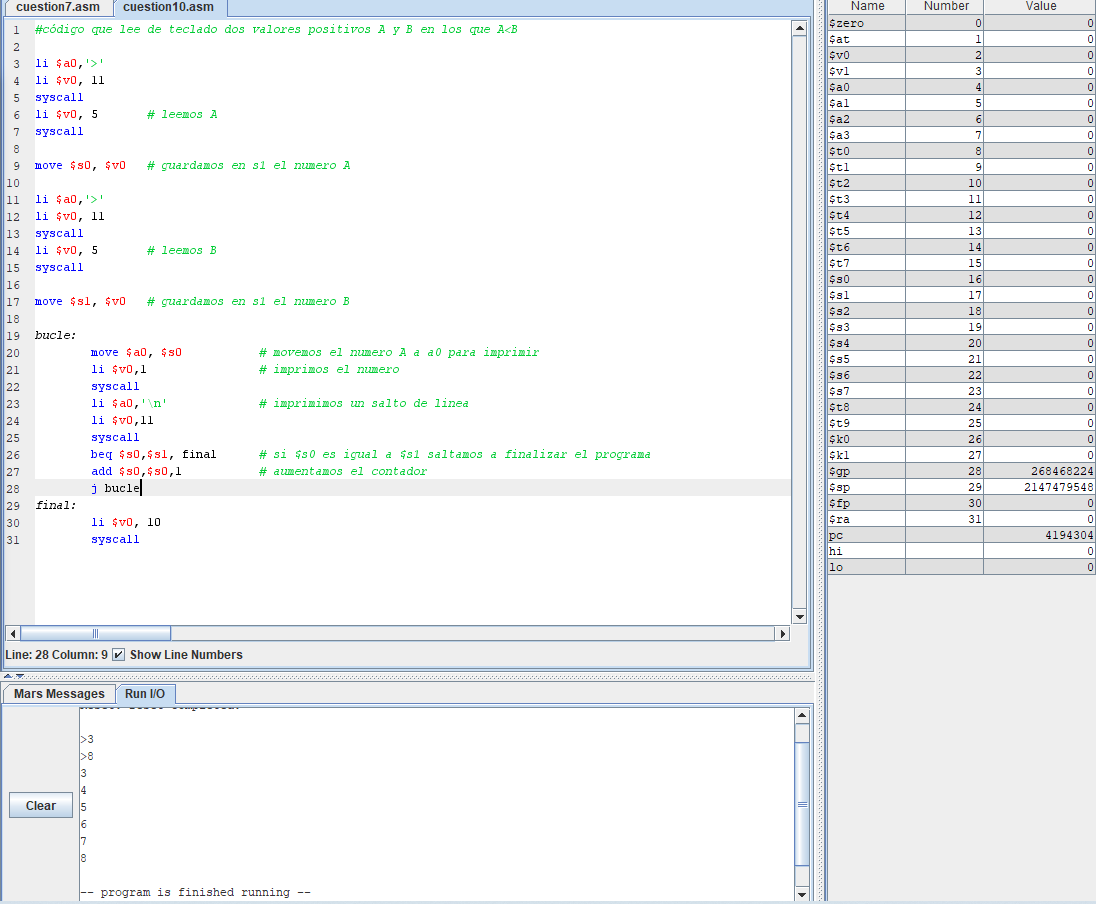
 Afegiu al codi de la qüestió 7 la possibilitat de llegir el valor n per teclat i escriure el resultat de la suma en la consola. Feu que hi haja un bucle infinit que llija pel teclat llevat que s’escriga el valor 0; en aquest cas s’acaba el programa.

Qüestió 9

 Feu el codi que llija dos enters de la consola i n’escriga la suma i torne a començar si el resultat de la suma és diferent de 0. El pseudocodi seria:

Qüestió 10

 Feu el codi que llig de teclat dos valors positius A i B en els quals A<B. El programa ha d’escriure per consola els valors compresos entre ells inclosos ella mateixa. Es a dir, si A=3 i B=6, escriu en la consola 3 4 5 6 (podeu escriure, per exemple, un salt de línia després de cadascun dels valors a mostrar).



Solicitamos los dos números y dentro de un bucle movemos el numero A (que lo utilizamos como contador) a $a0 para imprimirlo nada más comenzar, después tenemos un salto de línea. Ahora es cuando con la instrucción beq comprobamos si tenemos que terminar el bucle o no, miramos si $s0(contador) es igual a $s1(limite del contador), si lo es salta a final y termina el programa. Si no lo es, pasa a la siguiente instrucción que aumenta en 1 el contador y utiliza la instrucción de salto para volver al principio de bucle.