Chris Carr

Manchester Metropolitian University

1CWk50 – Task 2

**Task B**

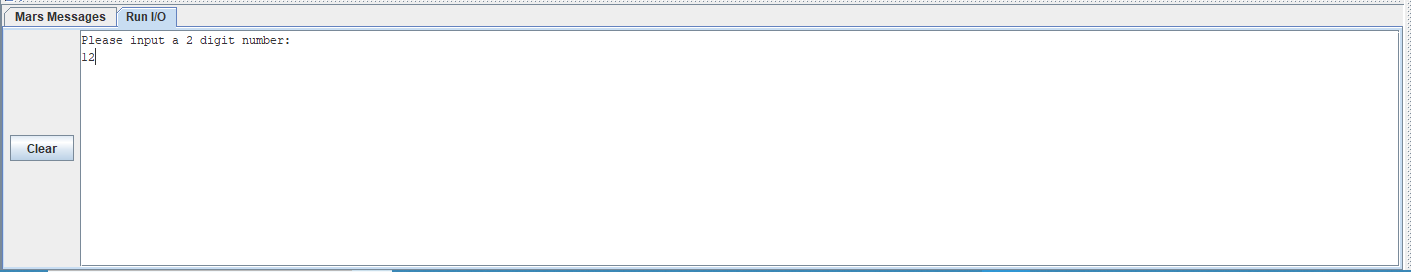
Task B introduced me to functions. I didn’t know how to correctly round in MiPS, so I checked the mfhi register to see if it contained any values, which meant that the number needed rounding up. The number would only ever need rounding up, due to the fact that the division was by 2 – the remainder would always be .5. This made it easy as all I did was add 1 to the output of the division.

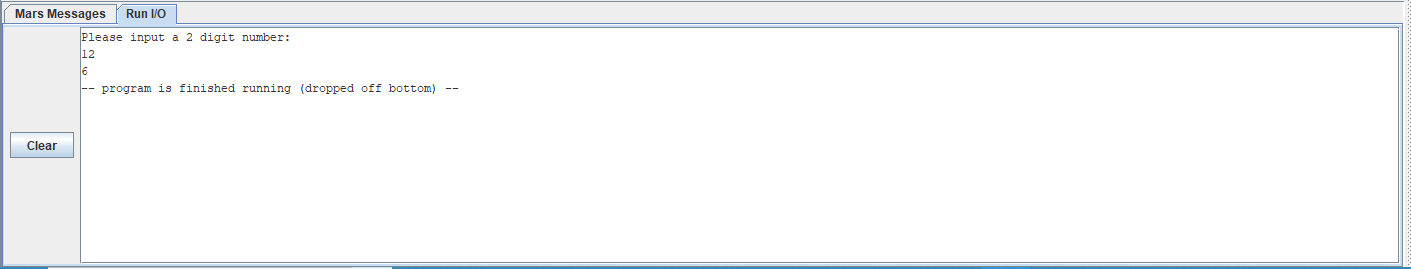
I noticed multiple functions which performed division, however I decided that the standard “div” function would suit my needs.

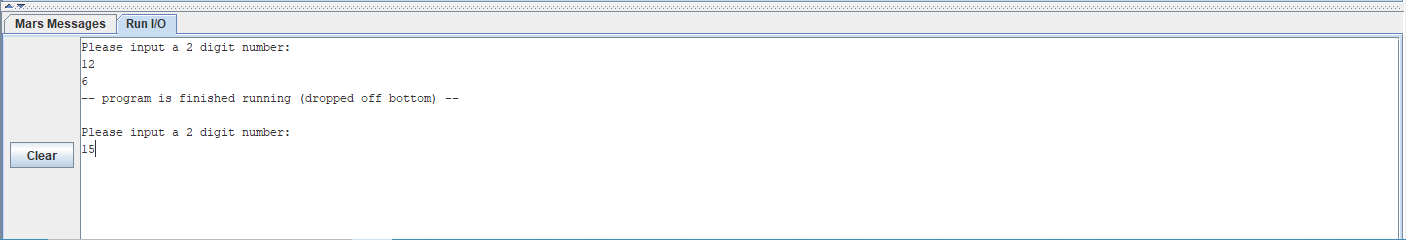
I may have used too many variables in this assignment. If I wanted to improve this program, my first step would be to reduce the number of variables used, and repeatedly use $a0 more often.

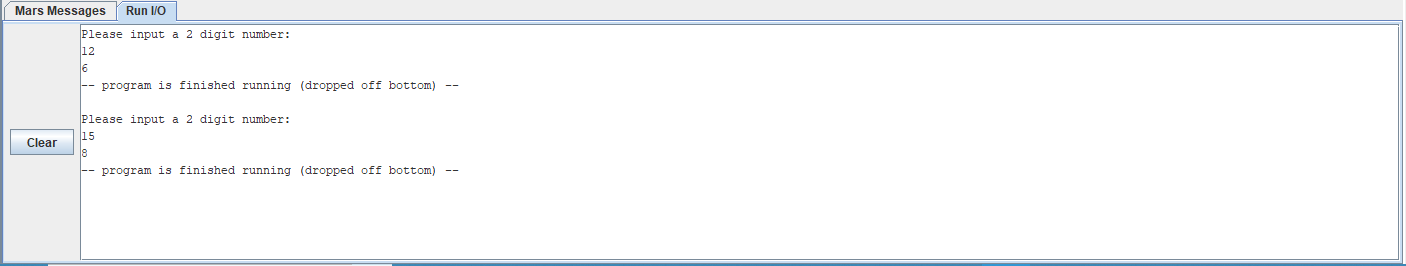
I also noticed how much my code is improving compared to the previous task – specifically using la instead of add, to move variables to where I want. I’m not 100% sure but I think la is more efficient than add. In the assemble process, it shows that la is converted into addi – not sure if using addi would be more efficient than la though.

*Screenshots*









*Code*

**.data**

**question:** **.asciiz** "Please input a 2 digit number: \n"

**.text**

#Display Question

**la** $a0, **question**

**li** $v0, 4

**syscall**

#Store user input

**li** $v0, 5

**syscall**

**la** $a1, ($v0)

#divide input by 2

**div** $a2, $a1, 2

#Check if integer needs to be rounded up (mfhi contains any remainder), branch to relevant function

**mfhi** $a3

**bgtz** $a3, **round**

**beqz** $a3, **output**

#round integer result (we cheat and just add 1)

**round:**

**add** $a2, $a2, 1

#output integer result

**output:**

**la** $a0, ($a2)

**li** $v0, 1

**syscall**

#end

Github commit: https://github.com/jynxmagic/MIPS-Assembly-Language/commit/e31b076fcf6ed5aa7b284679552d5a98f4d49dc3