Week 8 Lab - Part 8.1

Exercise 8.1.1

We're going to build our game up iteratively, and to start with, consider only 1 player. Lets start with some basic output to tell the player how many matchsticks are left (initially 15). We want the output to read "15 remaining" In ARMlite, write the code needed to do this.

If you need some further guidance, consider implementing this in the following steps):

1. create a label for the ASCII text (i.e., type . ASCIZ) you want to display (i.e, " remaining")
2. initialise register R0 to 15
3. write the value inside R0 to the output display
4. write the string " remaining"

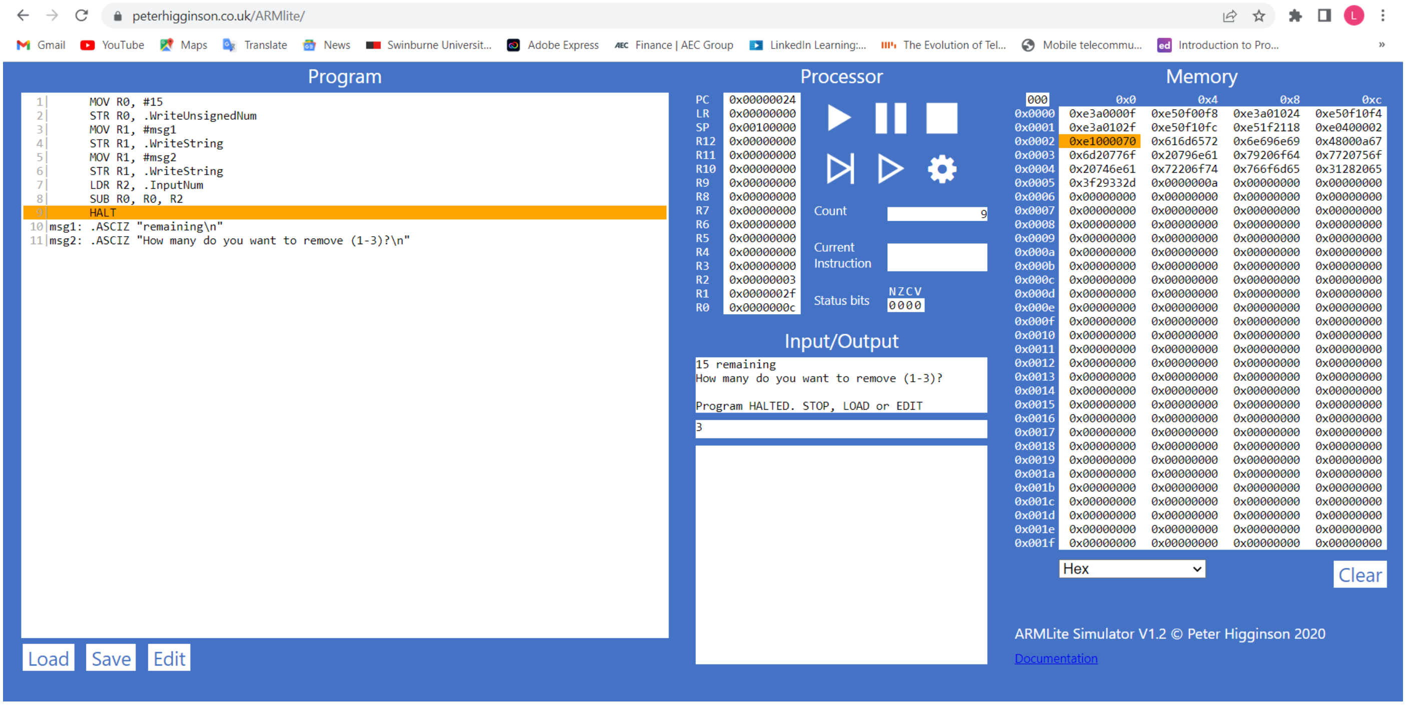
Exercise 8.1.2

Now lets consider getting some input from the user. Recall each player needs to provide a number between 1 and 3 (ie., the number of match sticks to remove). Building on what you implemented in 8.1.1, write the assembly code required to prompt the user for input using the string "How many do you want to remove (1-3)?", and then read in an integer value and store it in a register of your choice (but not one already being used!)

Hint - as in 8.1.1, define a label to store the string you want to display, and recall from this week's video lectures how numbers can be read in using LDR

Exercise 8.1.3

So now we have the player's input: the number of matchsticks to remove. We now need to calculate how many matchsticks remain once this number is removed. Again building on the code you wrote in the previous two exercises, write the code required to calculate the remaining number of matchsticks, and store this number into R0 (which recall, you initialised to 15 in 8.1.1).



```
      MOV R0, #15
      STR R0, .WriteUnsignedNum
      MOV R1, #msg1
      STR R1, .WriteString
      MOV R1, #msg2
      STR R1, .WriteString
      LDR R2, .InputNum
      SUB R0, R0, R2
      HALT
msg1: .ASCIZ "remaining\n"
msg2: .ASCIZ "How many do you want to remove (1-3)?\n"
```

```
PC   0x00000024
LR   0x00000000
SP   0x00100000
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000000
R8   0x00000000
R7   0x00000000
R6   0x00000000
R5   0x00000000
R4   0x00000000
R3   0x00000000
R2   0x00000003
R1   0x0000002f
R0   0x0000000c
```

Count 9

Current Instruction

Status bits NZCV 0000

```
15 remaining
How many do you want to remove (1-3)?

Program HALTED. STOP, LOAD or EDIT
```

```
3
```

```
000            0x0         0x4         0x8         0xc
0x0000  0xe3a0000f  0xe50f00f8  0xe3a01024  0xe50f10f4
0x0001  0xe3a0102f  0xe50f10fc  0xe51f2118  0xe0400002
0x0002  0xe1000070  0x616d6572  0x6e696e69  0x48000a67
0x0003  0x6d20776f  0x20796e61  0x79206f64  0x7720756f
0x0004  0x20746e61  0x72206f74  0x766f6d65  0x31282065
0x0005  0x3f29332d  0x0000000a  0x00000000  0x00000000
0x0006  0x00000000  0x00000000  0x00000000  0x00000000
0x0007  0x00000000  0x00000000  0x00000000  0x00000000
0x0008  0x00000000  0x00000000  0x00000000  0x00000000
0x0009  0x00000000  0x00000000  0x00000000  0x00000000
0x000a  0x00000000  0x00000000  0x00000000  0x00000000
0x000b  0x00000000  0x00000000  0x00000000  0x00000000
0x000c  0x00000000  0x00000000  0x00000000  0x00000000
0x000d  0x00000000  0x00000000  0x00000000  0x00000000
0x000e  0x00000000  0x00000000  0x00000000  0x00000000
0x000f  0x00000000  0x00000000  0x00000000  0x00000000
0x0010  0x00000000  0x00000000  0x00000000  0x00000000
0x0011  0x00000000  0x00000000  0x00000000  0x00000000
0x0012  0x00000000  0x00000000  0x00000000  0x00000000
0x0013  0x00000000  0x00000000  0x00000000  0x00000000
0x0014  0x00000000  0x00000000  0x00000000  0x00000000
0x0015  0x00000000  0x00000000  0x00000000  0x00000000
0x0016  0x00000000  0x00000000  0x00000000  0x00000000
0x0017  0x00000000  0x00000000  0x00000000  0x00000000
0x0018  0x00000000  0x00000000  0x00000000  0x00000000
0x0019  0x00000000  0x00000000  0x00000000  0x00000000
0x001a  0x00000000  0x00000000  0x00000000  0x00000000
0x001b  0x00000000  0x00000000  0x00000000  0x00000000
0x001c  0x00000000  0x00000000  0x00000000  0x00000000
0x001d  0x00000000  0x00000000  0x00000000  0x00000000
0x001e  0x00000000  0x00000000  0x00000000  0x00000000
0x001f  0x00000000  0x00000000  0x00000000  0x00000000
```

Hex

ARMLite Simulator V1.2 © Peter Higginson 2020

Week 8 Lab - Part 8.2

What happens if you enter a number that takes the number of matchsticks remaining beyond 0 (i.e., into negative values)? What do you think is going on here? Hint - take a look at the value in the register!

The value becomes more than a million as the 15 values are subtracted.

Question 8.2.2(a) - What is the condition that needs to be satisfied in order for this loop to occur? Write this as a comparison using an inequality (ie., less than, greater than, less than or equal, greater than or equal)

The register will be compared to a given number and the value in that register will be kept on subtracted and compared until the 15 is over.

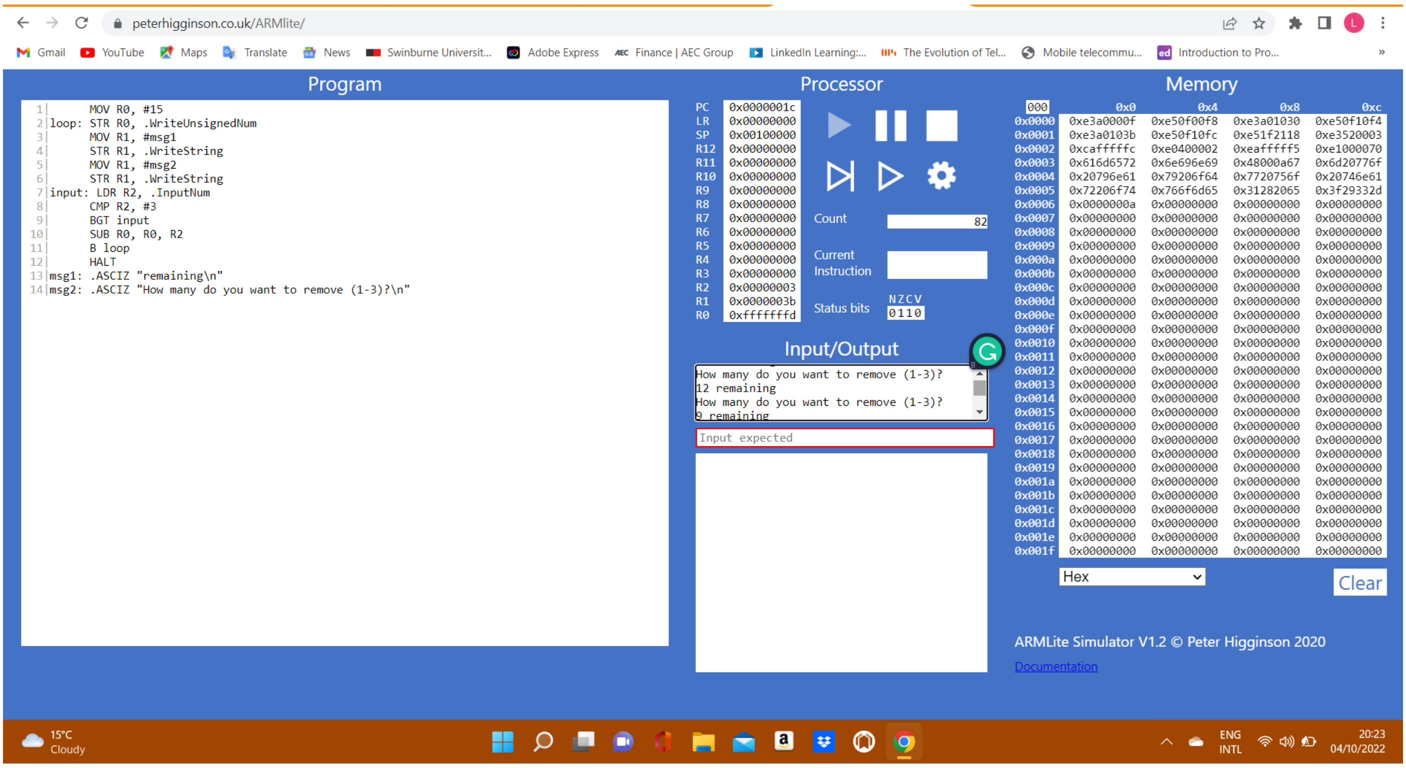
Question 8.2.2(b) - What two ARM assembly instructions could be used to create a branch that only occurs under this condition?

CMP: comparing the value in the register to a given number.

BGT: Bigger than, a condition is being put and the code will keep on looping and branching.

Question 8.2.2(c) - Based on the instructions you outlined in 8.2.2(b), what status bit would be set to 1 if the loop was to repeat? N

Question 8.2.2(d) - What are all the modifications needed to the current program to implement this feature ? Make the required modifications to your program to perform the task



```
      MOV R0, #15
loop: STR R0, .WriteUnsignedNum
      MOV R1, #msg1
      STR R1, .WriteString
      MOV R1, #msg2
      STR R1, .WriteString
input: LDR R2, .InputNum
      CMP R2, #3
      BGT input
      SUB R0, R0, R2
      B loop
      HALT
msg1: .ASCIZ "remaining\n"
msg2: .ASCIZ "How many do you want to remove (1-3)?\n"
```

```
PC   0x0000001c
LR   0x00000000
SP   0x00100000
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000000
R8   0x00000000
R7   0x00000000
R6   0x00000000
R5   0x00000000
R4   0x00000000
R3   0x00000000
R2   0x00000003
R1   0x0000003b
R0   0xfffffffd

Count 82
Status bits NZCV 0110
```

```
How many do you want to remove (1-3)?
12 remaining
How many do you want to remove (1-3)?
9 remaining
```

Input expected

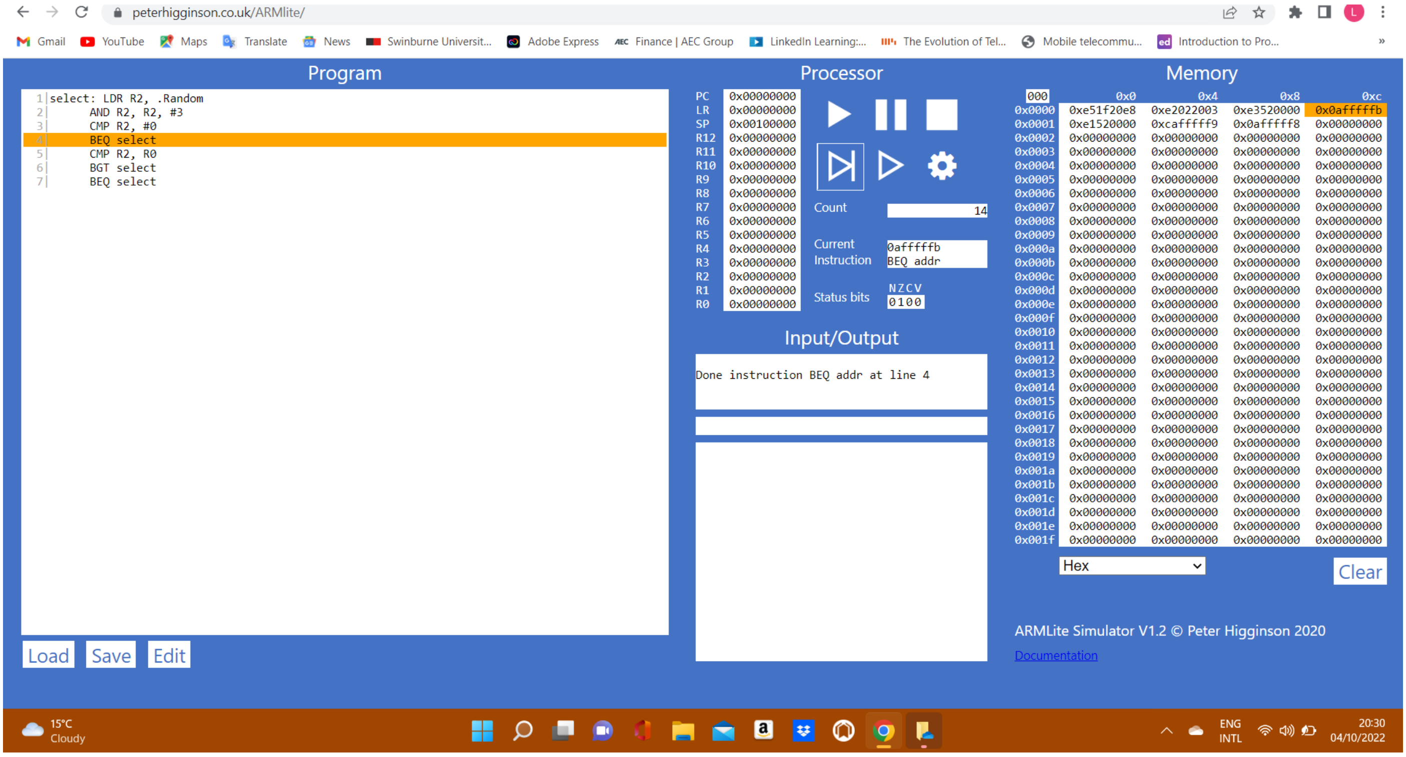
```
000           0x0         0x4         0x8         0xc
0x0000  0xe3a0000f  0xe50f00f8  0xe3a01030  0xe50f10f4
0x0001  0xe3a0103b  0xe50f10fc  0xe51f2118  0xe3520003
0x0002  0xcafffffc  0xe0400002  0xeafffff5  0xe1000070
0x0003  0x616d6572  0x6e696e69  0x48000a67  0x6d20776f
0x0004  0x20796e61  0x79206f64  0x7720756f  0x20746e61
0x0005  0x72206f74  0x766f6d65  0x31282065  0x3f29332d
0x0006  0x0000000a  0x00000000  0x00000000  0x00000000
0x0007  0x00000000  0x00000000  0x00000000  0x00000000
0x0008  0x00000000  0x00000000  0x00000000  0x00000000
0x0009  0x00000000  0x00000000  0x00000000  0x00000000
0x000a  0x00000000  0x00000000  0x00000000  0x00000000
0x000b  0x00000000  0x00000000  0x00000000  0x00000000
0x000c  0x00000000  0x00000000  0x00000000  0x00000000
0x000d  0x00000000  0x00000000  0x00000000  0x00000000
0x000e  0x00000000  0x00000000  0x00000000  0x00000000
0x000f  0x00000000  0x00000000  0x00000000  0x00000000
0x0010  0x00000000  0x00000000  0x00000000  0x00000000
0x0011  0x00000000  0x00000000  0x00000000  0x00000000
0x0012  0x00000000  0x00000000  0x00000000  0x00000000
0x0013  0x00000000  0x00000000  0x00000000  0x00000000
0x0014  0x00000000  0x00000000  0x00000000  0x00000000
0x0015  0x00000000  0x00000000  0x00000000  0x00000000
0x0016  0x00000000  0x00000000  0x00000000  0x00000000
0x0017  0x00000000  0x00000000  0x00000000  0x00000000
0x0018  0x00000000  0x00000000  0x00000000  0x00000000
0x0019  0x00000000  0x00000000  0x00000000  0x00000000
0x001a  0x00000000  0x00000000  0x00000000  0x00000000
0x001b  0x00000000  0x00000000  0x00000000  0x00000000
0x001c  0x00000000  0x00000000  0x00000000  0x00000000
0x001d  0x00000000  0x00000000  0x00000000  0x00000000
0x001e  0x00000000  0x00000000  0x00000000  0x00000000
0x001f  0x00000000  0x00000000  0x00000000  0x00000000
```

Week 8 Lab - Part 8.3

<u>Question 8.3.1(a)</u> What bit-wise operation can we perform on the register holding the 32-bit pattern to set all bits in the register to zero except the least significant2 bits Write this as a single line of code.

AND

<u>Question 8.3.1(b)</u> Using a label named "`select:`" Write the code needed to repeatedly sample a random number (from . Random) until the value is in the range 1-3. For now, just write this as a separate program and test it.



```
select: LDR R2, .Random
      AND R2, R2, #3
      CMP R2, #0
      BEQ select
      CMP R2, R0
      BGT select
      BEQ select
```

```
PC   0x00000000
LR   0x00000000
SP   0x00100000
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000000
R8   0x00000000
R7   0x00000000
R6   0x00000000
R5   0x00000000
R4   0x00000000
R3   0x00000000
R2   0x00000000
R1   0x00000000
R0   0x00000000
```

Count 14

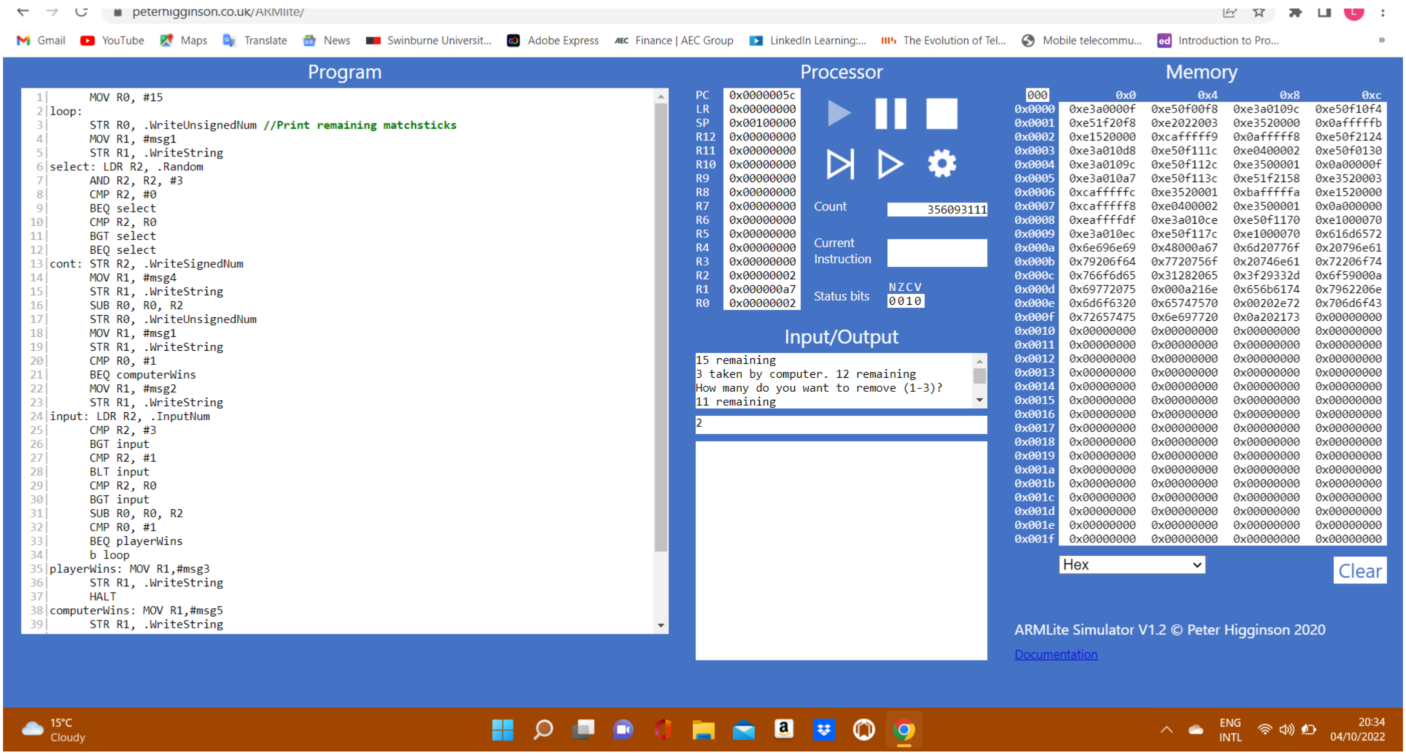
Current Instruction 0affffb BEQ addr

Status bits NZCV 0100

```
Done instruction BEQ addr at line 4
```

000	0x0	0x4	0x8	0xc
0x0000	0xe51f20e8	0xe2022003	0xe3520000	0x0affffb
0x0001	0xe1520000	0xcaffff9	0x0affff8	0x00000000
0x0002	0x00000000	0x00000000	0x00000000	0x00000000
0x0003	0x00000000	0x00000000	0x00000000	0x00000000
0x0004	0x00000000	0x00000000	0x00000000	0x00000000
0x0005	0x00000000	0x00000000	0x00000000	0x00000000
0x0006	0x00000000	0x00000000	0x00000000	0x00000000
0x0007	0x00000000	0x00000000	0x00000000	0x00000000
0x0008	0x00000000	0x00000000	0x00000000	0x00000000
0x0009	0x00000000	0x00000000	0x00000000	0x00000000
0x000a	0x00000000	0x00000000	0x00000000	0x00000000
0x000b	0x00000000	0x00000000	0x00000000	0x00000000
0x000c	0x00000000	0x00000000	0x00000000	0x00000000
0x000d	0x00000000	0x00000000	0x00000000	0x00000000
0x000e	0x00000000	0x00000000	0x00000000	0x00000000
0x000f	0x00000000	0x00000000	0x00000000	0x00000000
0x0010	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

Week 8 Lab - Part 8.4



```
      MOV R0, #15
loop:
      STR R0, .WriteUnsignedNum //Print remaining matchsticks
      MOV R1, #msg1
      STR R1, .WriteString
select: LDR R2, .Random
      AND R2, R2, #3
      CMP R2, #0
      BEQ select
      CMP R2, R0
      BGT select
      BEQ select
cont: STR R2, .WriteSignedNum
      MOV R1, #msg4
      STR R1, .WriteString
      SUB R0, R0, R2
      STR R0, .WriteUnsignedNum
      MOV R1, #msg1
      STR R1, .WriteString
      CMP R0, #1
      BEQ computerWins
      MOV R1, #msg2
      STR R1, .WriteString
input: LDR R2, .InputNum
      CMP R2, #3
      BGT input
      CMP R2, #1
      BLT input
      CMP R2, R0
      BGT input
      SUB R0, R0, R2
      CMP R0, #1
      BEQ playerWins
      b loop
playerWins: MOV R1,#msg3
      STR R1, .WriteString
      HALT
computerWins: MOV R1,#msg5
      STR R1, .WriteString
```

```
PC   0x0000005c
LR   0x00000000
SP   0x00100000
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000000
R8   0x00000000
R7   0x00000000
R6   0x00000000
R5   0x00000000
R4   0x00000000
R3   0x00000000
R2   0x00000002
R1   0x000000a7
R0   0x00000002
```

Count 356093111

Current Instruction

Status bits NZCV 0010

```
15 remaining
3 taken by computer. 12 remaining
How many do you want to remove (1-3)?
11 remaining
```

2

000	0x0	0x4	0x8	0xc
0x0000	0xe3a0000f	0xe50f00f8	0xe3a0109c	0xe50f10f4
0x0001	0xe51f20f8	0xe2022003	0xe3520000	0x0afffffb
0x0002	0xe1520000	0xcafffff9	0x0afffff8	0xe50f2124
0x0003	0xe3a010d8	0xe50f111c	0xe0400002	0xe50f0130
0x0004	0xe3a0109c	0xe50f112c	0xe3500001	0x0a00000f
0x0005	0xe3a010a7	0xe50f113c	0xe51f2158	0xe3520003
0x0006	0xcafffffc	0xe3520001	0xbafffffa	0xe1520000
0x0007	0xcafffff8	0xe0400002	0xe3500001	0x0a000000
0x0008	0xeaffffdf	0xe3a010ce	0xe50f1170	0xe1000070
0x0009	0xe3a010ec	0xe50f117c	0xe1000070	0x616d6572
0x000a	0x6e696e69	0x48000a67	0x6d20776f	0x20796e61
0x000b	0x79206f64	0x7720756f	0x20746e61	0x72206f74
0x000c	0x766f6d65	0x31282065	0x3f29332d	0x6f59000a
0x000d	0x69772075	0x000a216e	0x656b6174	0x7962206e
0x000e	0x6d6f6320	0x65747570	0x00202e72	0x706d6f43
0x000f	0x72657475	0x6e697720	0x0a202173	0x00000000
0x0010	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000