**Name**: Nguyen Quoc Thang

**ID** : 104193360

LAB 8 : SUBMISSION

**Excercise 8.1.1**

A screenshot of a computer

Description automatically generated

**Excercise 8.1.2** **A picture containing text, screenshot, software, computer icon

Description automatically generated**

**Excercise 8.1.3** **A screenshot of a computer

Description automatically generated**

**Excercise 8.2.1** A picture containing text, screenshot, software, computer icon

Description automatically generated

***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 of the R0 will be negative.

**Excercise 8.2.2**

**Question 8.2.2(a) - What is the condition that needs to be satisifed 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)**

* Satisfaction condition for the loop to occur is that the input value must not be between 1-3, it mean value entered must be < 1 and > 3

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

* BLT, BGT constructions could be used to create a branch that only occurs under this condition

**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 , Z , C status bit would be set to 1 if the loop was to repeat. N set to 1 when input value is 1 or 2 means values less than 3. Z set to 1 when input value is 3 and C set to one when we compare register with value 1.

**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.**

* I would add an iterator and use BLT to compare input value and value 1 and use BGT to compare input value and value 3 if the input value is less than 1 or greater than 3 then it will reverse the loop and fill in the value again from the beginning otherwise it will continue the program.

A screenshot of a computer

Description automatically generated

**Question 8.3.1(a)  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 signficant 2 bits ?  Write this as a single line of code.**

* **LSR R0, #30 *- (*** *It allows us to shift 30 bits of R0 to the right which means 30 bits of the value in R0 will become 0 only the last 2 bits will also be 2 bits random , it can be 0 or 1).*

*A computer screen with many different icons

Description automatically generated*

**Question 8.3.1(b)  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 seperate program and test it**

A screenshot of a computer

Description automatically generated with medium confidence

**Question 8.3.2(a)  -  Write the ARM assembly code that implements the algorithm expressed in the psuedo code above.   Implement this as a seperate stand alone program and initialise R0 with a number at the start of your program to allow you to test the functionality.  You wil want to test it using different values in R0.**

A screenshot of a computer

Description automatically generated with medium confidence

**Exercise 8.4**

***My code:***

1| MOV R0, #15

2|Loop1:

3| STR R0, .WriteSignedNum

4| MOV R1, #str

5| STR R1, .WriteString

6|Loop2:

7| MOV R2, #int

8| STR R2, .WriteString

9| LDR R2, .InputNum

10| STR R2, .WriteSignedNum

11| cmp R2, R0

12| BGT Loop2

13| cmp R2, #1

14| BLT Loop2

15| cmp R2, #3

16| BGT Loop2

17| B cont

18|cont:

19| SUB R0, R0, R2

20| STR R0, .WriteSignedNum

21| cmp R0 ,#0

22| BEQ result1

23| BGT select

24|select:

25| MOV R1, #str

26| STR R1, .WriteString

27| B comp

28|select2:

29| LDR R6, .Random

30| LSR R6, R6, #30

31| cmp R6, #0

32| BEQ select2

33| cmp R0, R6

34| BLT select2

35| SUB R0, R0, R6

36| cmp R0, #0

37| BEQ result

38| BGT Loop1

39|comp:

40| MOV R5, #compt

41| STR R5, .WriteString

42| B select2

43|result1:

44| MOV R4, #none

45| str R4, .WriteString

46| MOV R8, #rst

47| STR R8, .WriteString

48| B end

49|result:

50| MOV R4, #none

51| str R4, .WriteString

52| MOV R9, #rst1

53| STR R9, .WriteString

54|end:

55| HALT

56|str: .ASCIZ "remaining\n"

57|int: .ASCIZ "How many do you want to remove(1-3) ?\n"

58|none: .ASCIZ "There are no match stick\n"

59|compt: .ASCIZ "-----Computer turn-----"

60|rst: .ASCIZ "You lose\n"

61|rst1: .ASCIZ "You win\n"