CS1021 Tutorial #7
Using Memory

1 Set Intersection

Assume two mathematical *sets*, A and B, containing 32-bit values are stored in memory. Assume also that space in memory has been set aside to store a third set, C. The following ARM Assembler directives illustrate how the sets are arranged in memory.

```
ASize   DCD 8              ; Number of elements in Set A
AElems  DCD 7,20,9,17,3,2,23,13  ; Elements in Set A

Bsize   DCD 6              ; Number of elements in Set B
Belems  DCD 6,13,11,2,25,10      ; Elements of Set B

Csize   DCD 0              ; Number of elements in Set C
Celems  SPACE   56         ; Space for elements of Set C
```

Working in groups and beginning with a pseudo-code or Java solution, design and write a program that will create the third set, C, such that C is the *intersection* of A and B ($C = A \bigcap B$). (i.e. C should contain the elements that appear in both A and B.) In the example above $C = \{2, 13\}$. **Note that, by definition, elements should appear just once in a set.**

2 Unique Values

Working in groups, and beginning with a pseudo-code or Java solution, design and write an ARM Assembly Language program that will determine whether each word-size value in a list of ten word-size values in memory is unique (i.e. each value occurs only once in the list). If every value in the list is unique, your program should store the value 1 in R0, otherwise you should store 0 in R0.

For example, given the list below, your program should store a 1 in R0 because each value only occurs once.

5, 2, 7, 4, 13, 30, 18, 8, 9, 12

However, given the list below, your program should store a 0 in R0 because the value 4 occurs twice in the list.

5, 2, 7, 4, 13, 4, 18, 8, 9, 12

Assume that R1 contains the address of the first value in the list, that there are always ten values in the list, each value is one word in size and the values are stored contiguously in memory, as shown below:

```
start
    LDR r1, =VALUES ; list_addr = address of values

    <your program goes here>

stop    B   stop


    AREA    TestData, DATA, READWRITE

COUNT   EQU 10
VALUES  DCD 5, 2, 7, 4, 13, 4, 18, 8, 9, 12

    END
```