Austin Petersen
Ayden Dauenhauer
Lab 2
ELEN 120

Problem 1

1. What does this SUBS instruction do? SUBS r6, r6, #1

SUBS subtracts 1 from the value in register 6 and sets the flags

2. How many times are each of the statements in the loop executed when Array1 and Array2 match?

They are executed 10 times.

3. How many times is each of the following statements executed when the fourth element of Array 2 is changed to 100 from 11?

```
ADD r2, r2, #4 ; r2 points to next number in Array2
SUBS r6 r6, #1
BNE loop
MOV r5, #1
```

The statements are executed 3.

MOV r5, #1 does not get executed.

4. What is the result in r5 when the arrays are equal?

R5 has the value 0x1

5. What is the result in r5 when the arrays are different?

The value in R5 is 0xFFFFFFFF, which is #-1

Problem 2

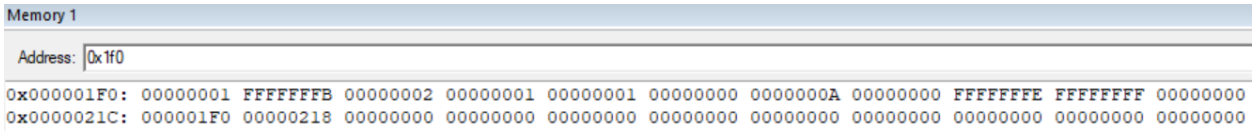
Before Values	0xC	0x11	0x5	0xB	0xFFFFFFFE	0xFFFFFFFF
After Values	0x12	0x16	0xA	0x10	0x3	0x4

main.s | startup_cm4.s

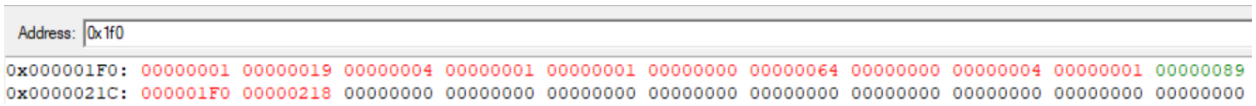
```
1         AREA main, CODE, READONLY
2         EXPORT __main
3         ENTRY
4  __main PROC
5
6         LDR r0, =Array1 ;initialize r0 with address of Array1
7         MOV r1, #12 ;count of numbers in array
8         MOV r5, #1 ;sets i variable
9         MOV r6, #7 ;number of loop iterations
10 loop   LDR r2, [r0]
11        CMP r5, r6 ; checks if r5 and r6 are equal
12        BEQ endless
13        ADDS r2, r2, #5 ;adds 5 to the 'i' element in array
14        ADDS r5, r5, #1 ;adds 1 to i to represent iteration
15        STR r2, [r0]
16        ADD r0, r0, #4 ; r1 points to next number in Array1
17        BNE loop
18 endless B endless
19        ENDP
20
21        ALIGN
22 Array1 DCD 12, 17, 5, 11, -2, -1, -4, 6, -3, 0, 69, 420 ; twelve 32-bit numbers
23
24        END
```

Problem 3:

Memory before the code running:



Memory after code running:



main.s | startup_cm4.s

```
1           AREA main, CODE, READONLY
2           EXPORT __main
3           ENTRY
4   __main PROC
5
6           LDR r0, =array ;initialize r0 with address of array
7           LDR r1, =result ;initializes r1 with the address of result
8           MOV r5, #0 ;sets i variable
9           MOV r6, #10 ;number of loop iterations
10  loop    LDR r2, [r0]
11          CMP r5, r6
12          BEQ endless
13          MUL r3, r2, r2 ;square of r2
14          ADD r4, r4, r3 ;summation
15          ADDS r5, r5, #1 ;adds 1 to i to represent iteration
16          STR r3, [r0]
17          STR r4, [r1]
18          ADD r0, r0, #4 ; r1 points to next number in Array1
19          BNE loop
20  endless B endless
21          ENDP
22
23          ALIGN
24  array   DCD 1, -5, 2, 1, 1, 0, 10, 0, -2, -1
25  result  DCD 0
26
27          END
```

Problem 4:

main.s | startup_cm4.s

```
1         AREA main, CODE, READONLY
2         EXPORT __main
3         ENTRY
4  __main PROC
5
6         LDR r1, =result ;initializes r1 with address of result
7         LDR r0, =num1 ;initialize r0 with address of num1
8         LDR r2, =num2 ;initialize r2 with address of num2
9         LDR r3, =num3 ;initialize r3 with address of num3
10        LDR r4, [r0] ;initialize r4 with value of num1
11        LDR r5, [r2] ;initialize r5 with value of num2
12        LDR r6, [r3] ;initialize r6 with value of num3
13        CMP r4, r5 ;compares r4 to r5
14        BLT part2 ;if r4 < r5, compare r4 with r6
15        CMP r5, r6 ;if r4 > r5, compare r5 with r6
16        BLT store5 ;if r5 < r6, store r1(result) with r5
17        STR r6, [r1] ;if r5 > r6, store r1(result) with r6
18
19 part2  CMP r4, r6 ;compares r4 to r6
20        BLT store4 ;if r4 < r6 store r4
21        STR r6, [r1] ;if r4 > r6 store r6
22        B endless
23
24 store4 STR r4, [r1] ;stores r4
25        B endless
26
27 store5 STR r5, [r1] ;stores r5
28
29 endless B endless
30        ENDP
31
32        ALIGN
33 num1   DCD 0x03247
34 num2   DCD 0x05431
35 num3   DCD 0x01120
36 result DCD 0
37        END
```

Memory 1

Address: 0x1fc

```
0x000001FC: 00001120 000001FC 000001F0 000001F4 000001F8
```

Problem 5:

main.s

```
1         AREA main, CODE, READONLY
2         EXPORT __main
3         ENTRY
4  __main PROC
5
6         LDR r0, =Array1 ;initialize r0 with address of Array1
7         MOV r1, #12 ;count of numbers in array
8         MOV r5, #1 ;sets i variable
9         MOV r6, #7 ;number of loop iterations
10 loop   LDR r2, [r0]
11        CMP r5, r6 ; checks if r5 and r6 are equal
12        BEQ endless
13        ADDS r2, r2, #5 ;adds 5 to the 'i' element in array
14        ADDS r5, r5, #1 ;adds 1 to i to represent iteration
15        STR r2, [r0]
16        ADD r0, r0, #4 ; r1 points to next number in Array1
17        BNE loop
18 endless B endless
19        ENDP
20
21        ALIGN
22 Array1 DCD 12, 17, 5, 11, -2, -1, -4, 6, -3, 0, 69, 420 ; twelve 32-bit numbers
23
24        END
```

Memory 1

Address: 0x1f8

```
0x000001F8: 00000000 00000001 00000001 00000002 00000003 00000005 00000008 0000000D 00000015 00000022 00000037
0x00000224: 00000059 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 000001F8 00000000
```