EEP3020 : Digital Systems Lab
Assignment - I

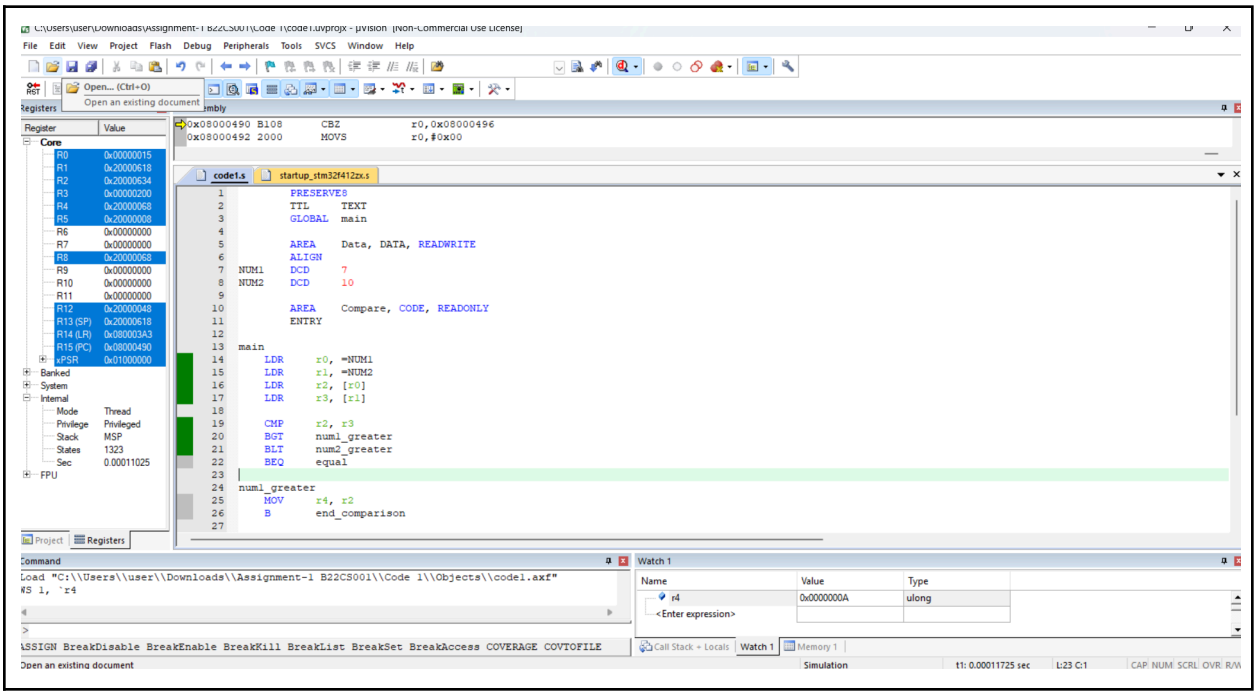
Name : *Abhay Kashyap*
Roll no : *B22CS001*

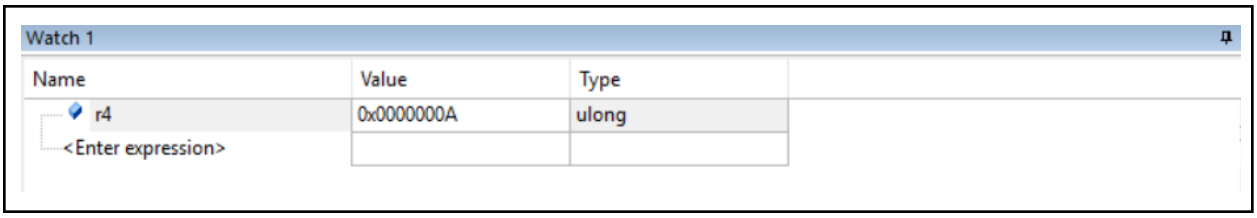
Question - 1. Check Greater between two input numbers

```
                 PRESERVE8
        TTL     TEXT
        GLOBAL  main

        AREA    Data, DATA, READWRITE
        ALIGN
NUM1    DCD     7
NUM2    DCD     10

        AREA    Compare, CODE, READONLY
        ENTRY

main
    LDR     r0, =NUM1
    LDR     r1, =NUM2
    LDR     r2, [r0]
    LDR     r3, [r1]

    CMP     r2, r3
    BGT     num1_greater
    BLT     num2_greater
    BEQ     equal

num1_greater
    MOV     r4, r2
    B       end_comparison

num2_greater
    MOV     r4, r3
    B       end_comparison

equal
    MOV     r4, r2

end_comparison
    MOV     r7, #1
    SWI     0

END
```

Explanation :

The ARM assembly code compares two numbers (7 and 10, stored in NUM1 and NUM2) and stores the greater of the two in register r4. If both numbers are equal, r4 is set to the value of NUM1. After the comparison, the program makes a system call to exit. The result of the comparison is controlled by conditional branching (BGT, BLT, BEQ).





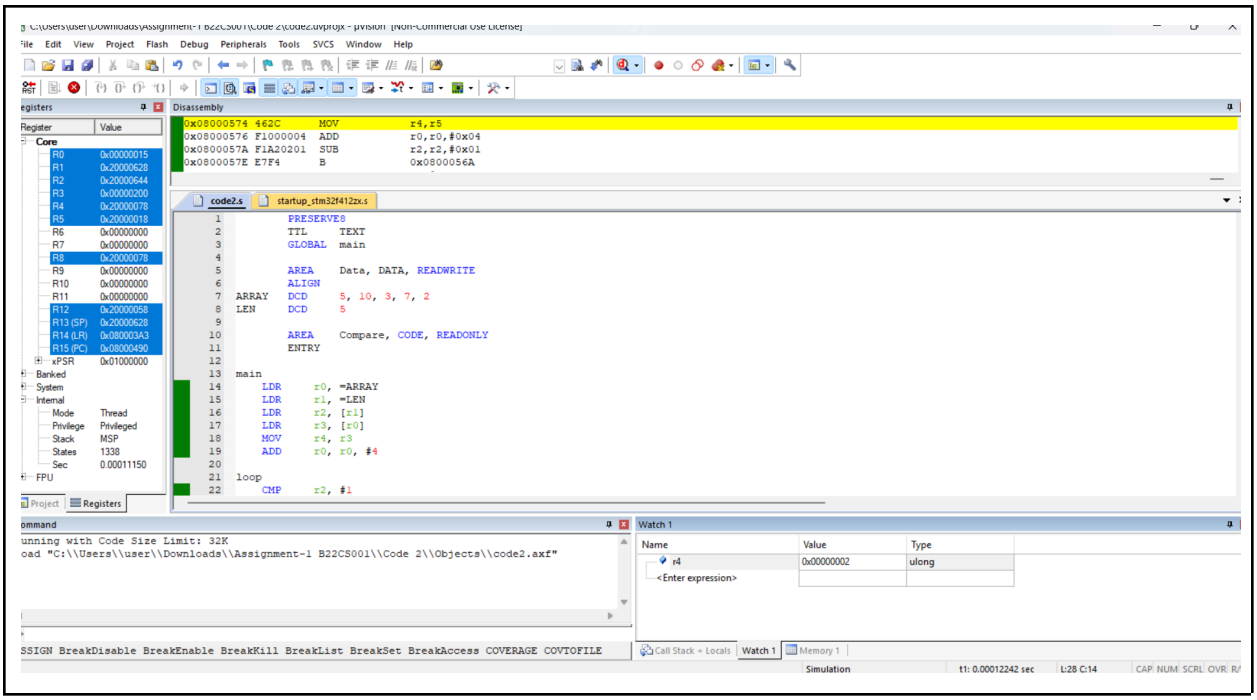
Question - 2. Calculate the minimum one between elements of an array

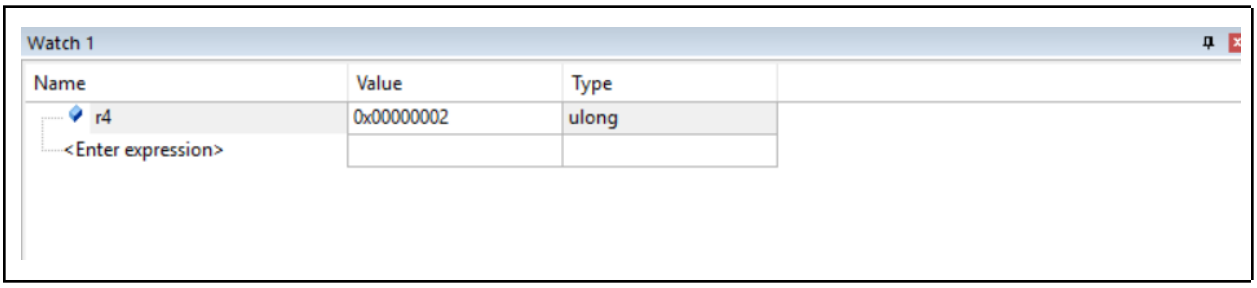
```
        PRESERVE8
        TTL     TEXT
        GLOBAL  main

        AREA    Data, DATA, READWRITE
        ALIGN
ARRAY   DCD     5, 10, 3, 7, 2
LEN     DCD     5
```

```
        AREA    Compare, CODE, READONLY
        ENTRY

main
    LDR     r0, =ARRAY
    LDR     r1, =LEN
    LDR     r2, [r1]
    LDR     r3, [r0]
    MOV     r4, r3
    ADD     r0, r0, #4

loop
    CMP     r2, #1
    BEQ     end_comparison

    LDR     r5, [r0]
    CMP     r5, r4
    BGE     skip_update
    MOV     r4, r5

skip_update
    ADD     r0, r0, #4
    SUB     r2, r2, #1
    B       loop

end_comparison
    MOV     r7, #1
    SWI     0

END
```

Explanation :

The ARM assembly code finds the minimum value in an array (ARRAY with values {5, 10, 3, 7, 2}) and stores it in register r4. It iterates through the array, comparing each element to the current minimum, and updating the minimum when a smaller value is found. The loop continues until all elements are checked, and then the program exits. The array has 5 elements, and its length is stored in LEN.





Question - 3. Check if a number is divisible by 7 from 5 input numbers

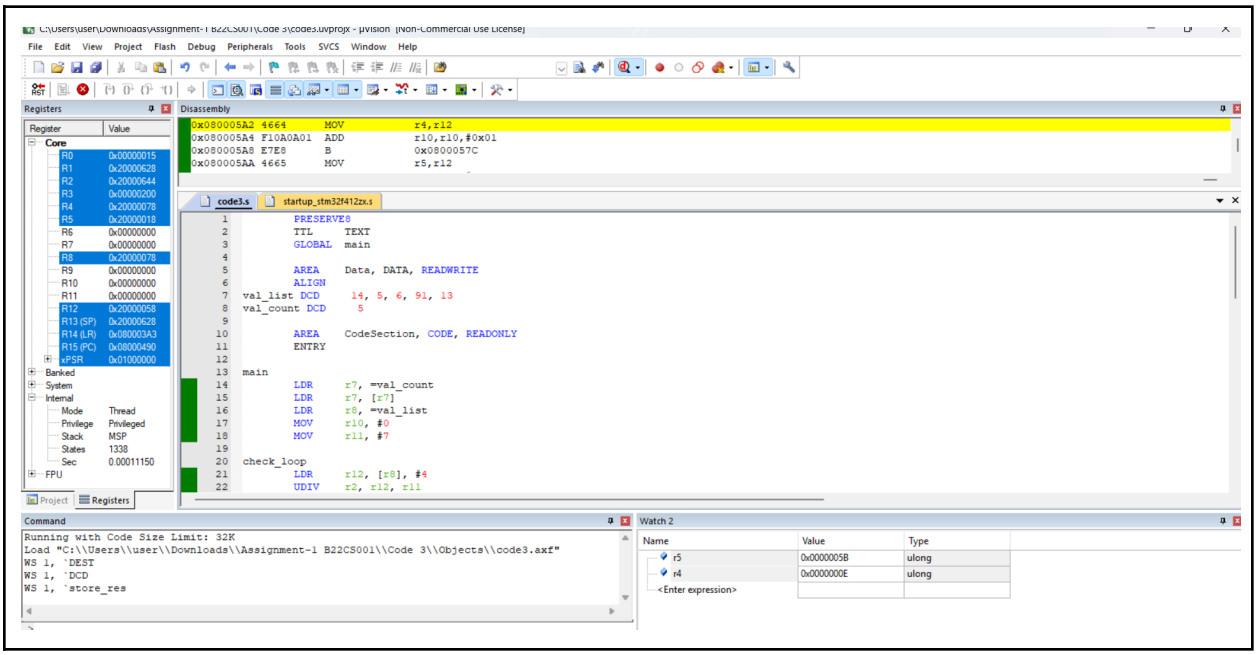
```
          PRESERVE8
        TTL      TEXT
        GLOBAL   main

        AREA     Data, DATA, READWRITE
        ALIGN
val_list DCD      14, 5, 6, 91, 13
val_count DCD      5

        AREA     CodeSection, CODE, READONLY
        ENTRY

main
        LDR      r7, =val_count
```

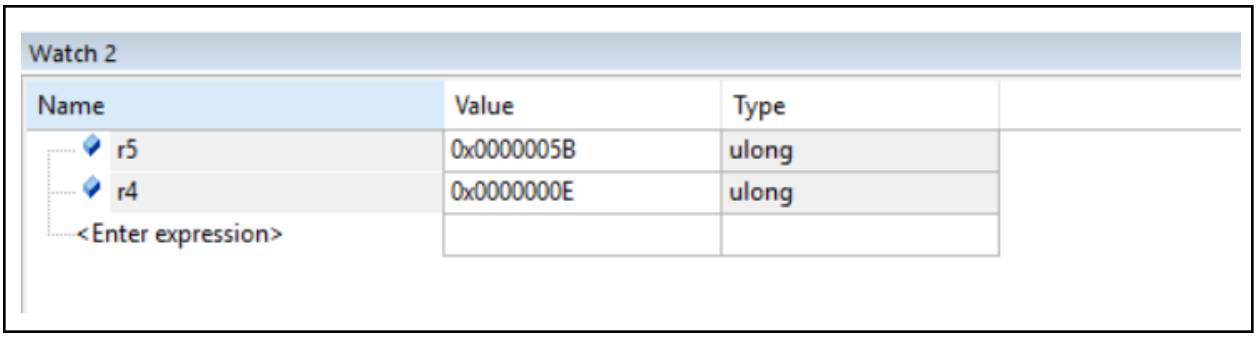
```
        LDR     r7, [r7]
        LDR     r8, =val_list
        MOV     r10, #0
        MOV     r11, #7

check_loop
        LDR     r12, [r8], #4
        UDIV    r2, r12, r11
        MLS     r2, r2, r11, r12
        CMP     r2, #0
        BEQ     save_val

next_loop
        SUBS    r7, r7, #1
        BNE     check_loop

end_loop
        B       end_loop

save_val
        CMP     r10, #0
        BEQ     store_r4
        CMP     r10, #1
        BEQ     store_r5
        CMP     r10, #2
        BEQ     store_r6
        CMP     r10, #3
        BEQ     store_r7
        CMP     r10, #4
        BEQ     store_r8
        B       next_loop

store_r4
        MOV     r4, r12
        ADD     r10, r10, #1
        B       next_loop

store_r5
        MOV     r5, r12
        ADD     r10, r10, #1
        B       next_loop

store_r6
        MOV     r6, r12
        ADD     r10, r10, #1
        B       next_loop

store_r7
        MOV     r7, r12
        ADD     r10, r10, #1
        B       next_loop
```

```
store_r8
        MOV     r8, r12
        ADD     r10, r10, #1
        B       next_loop

        END
```

Explanation :

The code checks each number in the val_list to see if it's divisible by 7. If a number is divisible, it stores the number in one of the registers (r4, r5, r6, r7, r8). The loop continues until all numbers are checked, and the program ends once all divisible numbers are saved. The loop counter (r7) keeps track of how many numbers remain to be processed.



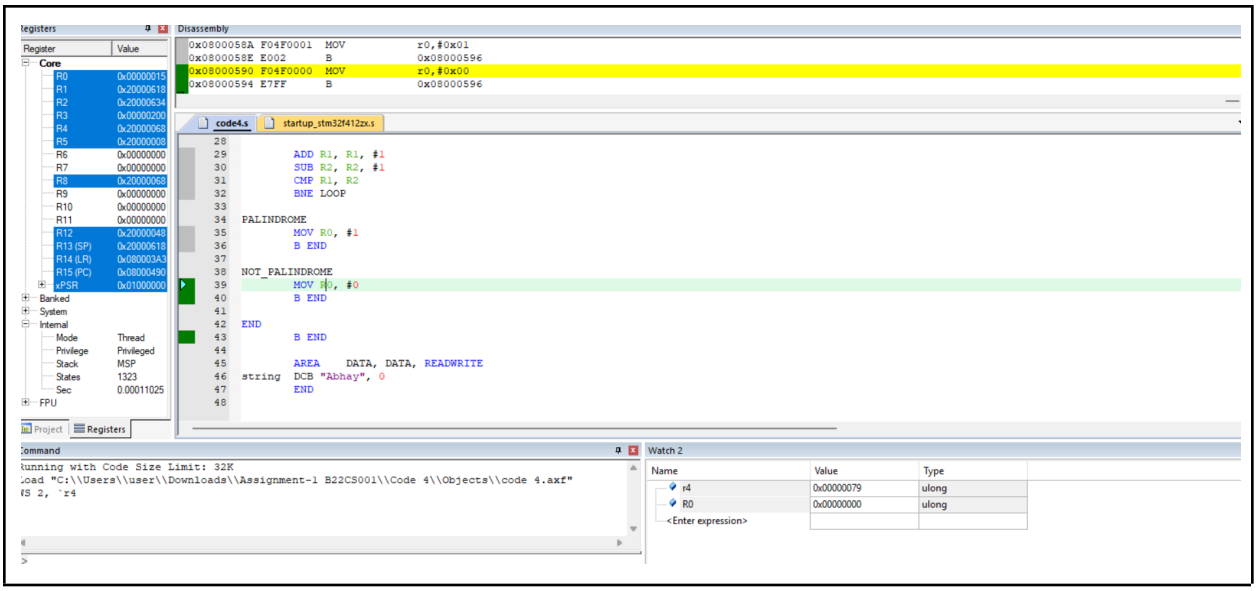


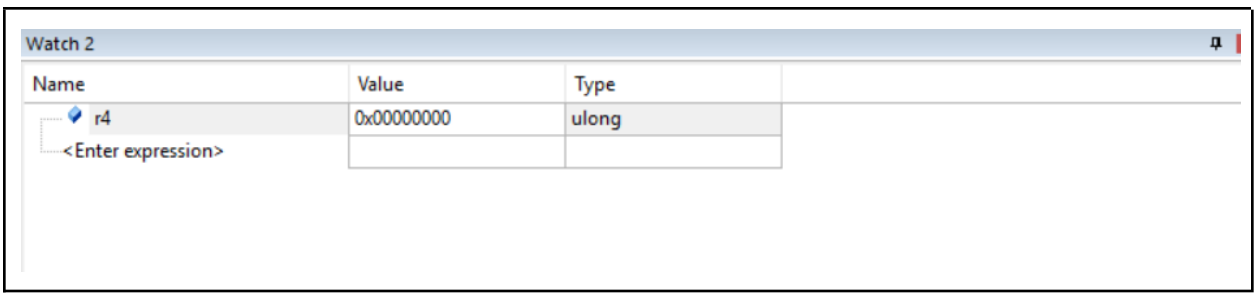
Question - 4. Check if a string is a palindrome or not

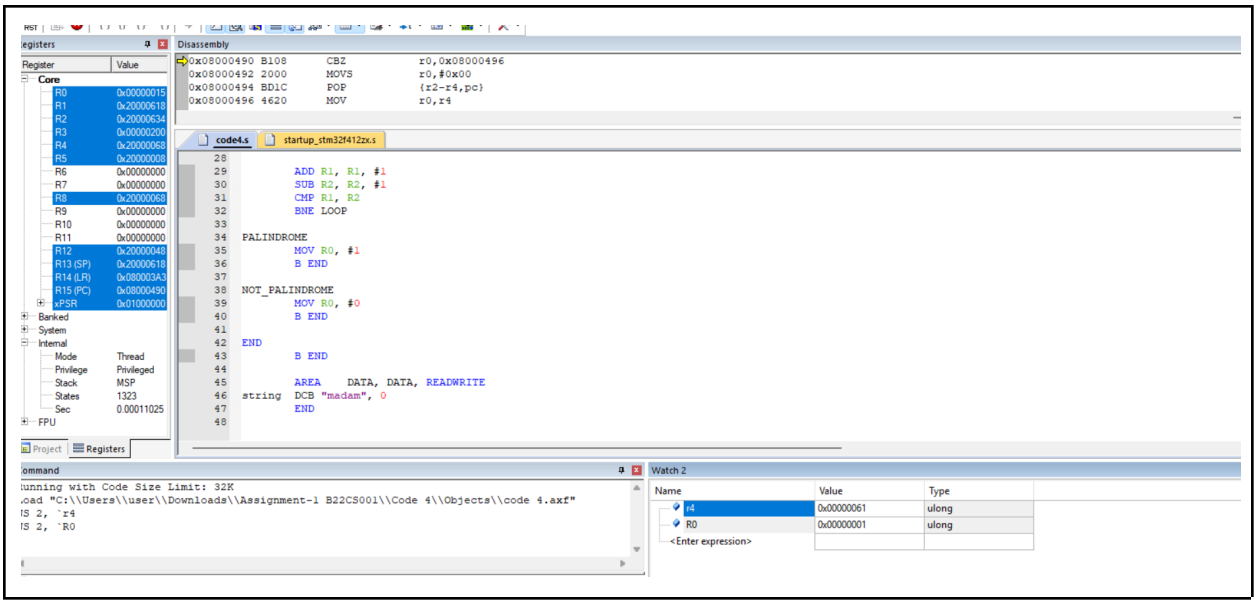
```
        PRESERVE8
        TTL     TEXT
        GLOBAL  main

        AREA    RESET, CODE, READONLY
        ENTRY

main
        LDR R0, =string
        MOV R1, #0
        MOV R2, #0

FIND_END
        LDRB R3, [R0, R2]
        CMP R3, #0
        BEQ BEGIN_COMPARISON
        ADD R2, R2, #1
        B FIND_END

BEGIN_COMPARISON
        SUB R2, R2, #1

LOOP
        LDRB R3, [R0, R1]
        LDRB R4, [R0, R2]
        CMP R3, R4
        BNE NOT_PALINDROME

        ADD R1, R1, #1
        SUB R2, R2, #1
        CMP R1, R2
        BNE LOOP

PALINDROME
        MOV R0, #1
        B END

NOT_PALINDROME
        MOV R0, #0
        B END

END
        B END

        AREA    DATA, DATA, READWRITE
string  DCB "Abhay", 0
        END
```

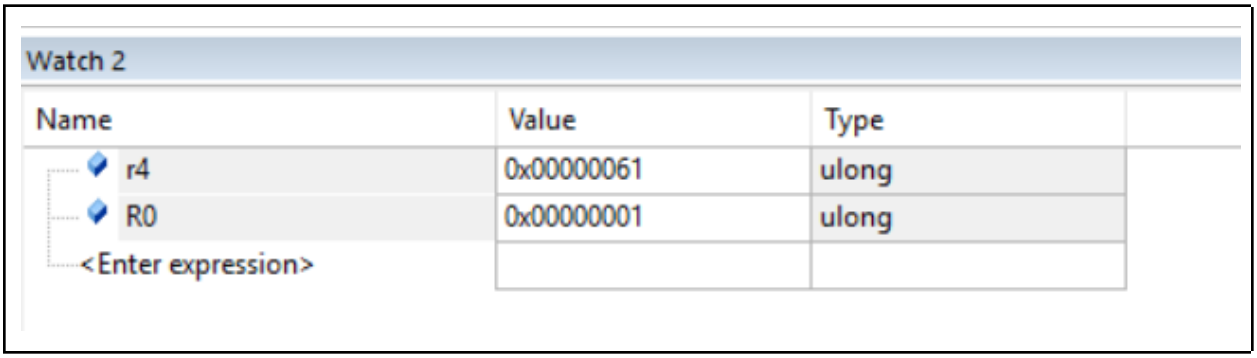
Explanation :

This ARM assembly code checks if a string is a palindrome by comparing characters from both ends towards the center. It first loads the string and calculates its length by finding the null terminator. Then, it compares corresponding characters from the left and right, updating indices until either a mismatch is found (resulting in R0 being set to 0 for not a palindrome) or all characters match (resulting in R0 being set to 1 for a palindrome).






Disassembly
0x08000490 B108 CBZ r0,0x08000496
0x08000492 2000 MOVS r0,#0x00
0x08000494 BD1C POP {r2-r4,pc}
0x08000496 4620 MOV r0,r4
code4.s
startup_stm32f412zx.s
ADD R1, R1, #1
SUB R2, R2, #1
CMP R1, R2
BNE LOOP
PALINDROME
MOV R0, #1
B END
NOT_PALINDROME
MOV R0, #0
B END
END
B END
AREA DATA, DATA, READWRITE
string DCB "madam", 0
END
Watch 2
Name
Value
Type
r4
0x00000061
ulong
R0
0x00000001
ulong
<Enter expression>


Watch 2
Name
Value
Type
r4
0x00000061
ulong
R0
0x00000001
ulong
<Enter expression>