

Practical 2

Branches and Loops

Prerequisite: students are required to have read the lecture notes from page 14 to page 26.

Step 1

- Without using the debugger, determine the values of **D1**, **D2**, **D3** and **D4** after the execution of the following loops.

```

                                org      $4
Vector_001  dc.l      Main

                                org      $500
Main
                                clr.l     d1
                                move.l    #$80000007,d0
loop1
                                addq.l    #1,d1
                                subq.w    #1,d0
                                bne       loop1

                                clr.l     d2
                                move.l    #$fe2310,d0
loop2
                                addq.l    #1,d2
                                subq.b    #2,d0
                                bne       loop2

                                clr.l     d3
                                moveq.l    #125,d0
loop3
                                addq.l    #1,d3
                                dbra      d0,loop3      ; DBRA = DBF

                                clr.l     d4
                                moveq.l    #10,d0
loop4
                                addq.l    #1,d4
                                addq.l    #1,d0
                                cmpi.l    #30,d0
                                bne       loop4

                                illegal

```

- Assemble and run the program above to check your answers.

Step 2

Let us consider the following program:

VALUE	equ	18
	org	\$4
Vector_001	dc.l	Main
	org	\$500
Main	move.b	#VALUE,d1
	tst.b	d1
	bne	next1
	move.l	#200,d0
	bra	quit
next1	bmi	next3
	cmp.b	#\$61,d1
	blt	next2
	move.l	#400,d0
	bra	quit
next2	move.l	#600,d0
	bra	quit
next3	move.l	#800,d0
quit	illegal	

This program loads a value into **D0.L** (the output register) according to the value of **D1.B** (the input register), which is initialized at the beginning of the source code with the VALUE label.

Answer the following questions without using the debugger.

1. What value is returned by the program when the VALUE label is set to 18?
2. What value is returned by the program when the VALUE label is set to -5?
3. What value is returned by the program when the VALUE label is set to 0?
4. What value is returned by the program when the VALUE label is set to 96?

Assemble and run the program above for each value of the VALUE label and check your answers.

Step 3

Write the **Abs** program that returns the absolute value of a signed integer.

Input : **D0.L** = 32-bit signed integer.

Output : **D0.L** = **|D0.L|**

Use the following structure in order to run and test your program (try several significant values for **D0**).

```

                org      $4
Vector_001     dc.l      Main
                org      $500
Main          move.l     #-1,d0      ; Initialize D0.
Abs           ; ...                ; Abs program.
              ; ...                ; Once executed, D0.L should hold
              ; ...                ; the absolute value of the input.

              illegal

```

Note: Have a look at the NEG instruction.

Step 4

Write the **StrLen** program that returns the length of a string (ending with a null character).

Input : **A0.L** points to a string whose length is to be found.

Output : **D0.L** returns the length of the given string (not including the null character).

Use the following structure in order to run and test your program:

```

                org      $4
Vector_001     dc.l      Main
                org      $500
Main          movea.l     #STRING,a0 ; A0 points to the string.
StrLen        ; ...                ; StrLen program.
              ; ...                ; Once executed, D0.L should hold
              ; ...                ; the length of the string.

              illegal

                org      $550
STRING        dc.b       "This string is made up of 40 characters.",0

```

Note: In order to avoid encoding problems, do not use accented characters.

Find where the string is located by using the **[Mémoire]** tab.

Step 5

Write the **SpaceCount** program that returns the number of spaces in a string (ending with a null character).

Input : **A0.L** points to a string whose number of spaces is to be found.

Output : **D0.L** returns the number of spaces in the given string.

Use the following structure in order to run and test your program:

```

                                org      $4
Vector_001  dc.l      Main
                                org      $500
Main        movea.l #STRING,a0 ; A0 points to the string.
SpaceCount  ; ...              ; SpaceCount program.
            ; ...              ; Once executed, D0.L should hold
            ; ...              ; the number of spaces in the string.

            illegal
                                org      $550
STRING      dc.b      "This string contains 4 spaces.",0

```

Tip: To get the ASCII code of the space character, you can use the following syntax: #' '.

Note: In order to avoid encoding problems, do not use accented characters.

Step 6

Write the **LowerCount** program that returns the number of small letters in a string (ending with a null character).

Input : **A0.L** points to a string whose number of small letters is to be found.

Output : **D0.L** returns the number of small letters in the given string.

Use the following structure in order to run and test your program:

```

                                org      $4
Vector_001  dc.l      Main
                                org      $500
Main        movea.l #STRING,a0 ; A0 points to the string.
LowerCount  ; ...              ; LowerCount program.
            ; ...              ; Once executed, D0.L should hold
            ; ...              ; the number of small letters in the string.

            illegal
                                org      $550
STRING      dc.b      "This string contains 29 small letters.",0

```

Tips:

- To get the ASCII code of the *a* character, you can use the following syntax: `#'a'`.
- To get the ASCII code of the *z* character, you can use the following syntax: `#'z'`.
- A character is a small letter if its ASCII code ranges from *a* to *z*.

Note: In order to avoid encoding problems, do not use accented characters.