Assembler directives - cseg, rseg, equ, end

CSEG	111	CSEG [AT *absolute address*]	Define an absolute segment within the code address space.
RSEG	110	RSEG *seg*	Select a relocatable segment.
SEGMENT	106	*seg* SEGMENT *class* [*reloctype*] [*alloctype*]	Define a relocatable segment.
USING	134	USING *expression*	Set the predefined symbolic register address and reserve space for the specified register bank.
EQU	113	EQU *expression*	Set symbol value permanently.

SEG directives are used to locate program code, constant data, and variables in specific places in memory.

Example: Blink program. Note that program execution always starts on reset at location 0000. A long jump instruction is usually placed there so that the actual program can be placed anywhere in code memory.

```
; Reset Vector
cseg AT 0
ljmp Main             ; Locate a jump to the start of code
at
                      ; the reset vector.

;-------------------------------------------------------------
------------------
; CODE SEGMENT
;-------------------------------------------------------------
------------------

Blink segment CODE     ; Define Blink as a segment of code.
      rseg   Blink     ; Switch to this code segment.
      using 0          ; Specify register bank for the
following
                       ; program code.
```

EQU is used to define symbols used in the code. Note the notation below that can be used for individual bits of the Port 1 register.

For example:

```
GREEN_LED equ P1.6     ; Port I/O pin connected to Green
LED.
```

Later in the code, we can refer to the symbol:

```
clr GREEN_LED
```

Exercise:

Modify the reg_mem program as follows:

Start the "Main" routine at location 2000h using a cseg assembler directive.

Use an EQU assembler directive to define the constant (25h) that was moved around in this program.

Assemble the code. Open the **reg_mem.lst** file. Verify that the Main program now starts at location 2000h.