Create an array in RAM named **Numbers** with the following 16bit wide elements:

-2,000  -1,000  -500  -500  -400  -600  -1600  -1400

Create a 16bit wide variable in RAM with the name **x**

Write an assembly language program which calculates the average of all the elements of the array and stores the average value in the variable named **x**.

**Use loops for repeated tasks. Include pseudo code and flow charts (in pdf format) and the assembly file (.asm format) in your package. Include screen shots of the memory browser which show the elements of the array as well as the variable "**x**" clearly showing the average value of the array stored in it. Use the Carmen dropbox setup for this MT.**
Pseudo Code

sum = 0;
For(i = 0; i < 16, i=i+2)
{
    sum = sum + Numbers[i];
}

// Divide Sum by 8 = 2^3 by using bit shifting
For(j=0, j< 3, ++j)
{
    Roll Right sum by one bit;
}

x = sum;
Flow Chart

```
SUM = 0
I = 0

cond_label_for1:

break_for1: Y
i > 16
N

SUM = SUM + Numbers[i]

i = i + 2

break_for1:

j = 0
cond_label_for2

break_for2: Y
j > 3
N

rra Sum

++j
cond_label_for2

break_for2:

x = Sum
```
Code

;---------------------------------------------------------------
; MSP430 Assembler Code Template for use with TI Code Composer Studio
;
;---------------------------------------------------------------
.cdecls C,LIST, "msp430.h" ; Include device header file

;---------------------------------------------------------------
.data
Numbers: .word -2000, -1000, -500, -500, -400, -600, -1600, -1400
x: .space 2
sum: .space 2
;---------------------------------------------------------------
.text ; Assemble into program memory
.retain ; Override ELF conditional linking
; and retain current section
.retainrefs ; Additionally retain any sections
; that have references to current
; section
;---------------------------------------------------------------
RESET mov.w #__STACK_END, SP ; Initialize stack pointer
StopWDT mov.w #WDTPW|WDTHOLD,&WDTCTL ; Stop watchdog timer

;---------------------------------------------------------------
; Main loop here
;---------------------------------------------------------------
    mov.w #0, &sum ; Actually this statement is not needed
    ; since Sum is already initialized to 0
    mov.w #0, R10 ; R10 is index i

cond_label_for1:

    cmp.w #16, R10
    jge break_for1
    add.w Numbers(R10), &sum
    incd.w R10
    jmp cond_label_for1

break_for1:

... continued on next page
... continued from previous page

```assembly
mov.w #0, R11          ; R11 is index j
cond_label_for2:
cmp.w #3, R11
jge break_for2
rra.w &sum
inc.w R11
jmp cond_label_for2
break_for2:

mov.w &sum, &x

loop: jmp loop

;----------------------------------------------------------------------------------------
;                      Stack Pointer definition
;----------------------------------------------------------------------------------------
.global __STACK_END
.sect .stack

;----------------------------------------------------------------------------------------
;                     Interrupt Vectors
;----------------------------------------------------------------------------------------
.sect ".reset"        ; MSP430 RESET Vector
.short RESET
```

**Screenshot of the Memory Browser**

![Memory Browser Screenshot](image-url)