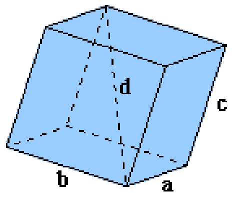
CS 218 – Assignment #5

Purpose:	Learn to use arithmetic instructions, control instructions, compare instructions, and conditional jump instructions.
Due:	Tuesday (6/14)
Points:	80

Assignment:

Write a simple assembly language program to calculate the some geometric information for a series of cubes; areas and volumes. The information for the cubes are stored in an array. Once the areas and volumes are computed, the program should find the minimum, maximum, middle value, sum, and average for the areas and volumes.



The formulas for cube area and volume are as follows:

$$\boldsymbol{cubeAreas[i] = 6 * sides[i]^2}$$
$$\boldsymbol{cubeVolumes[i] = sides[i]^3}$$

Where ***i*** represents the specific cube information in the array (i.e., the index).

The side lengths are stored in a word-sized array named ***sides***. The cube areas must be calculated and stored into the word-sized array named ***cubeAreas***. The areas sum, ***caSum***, must be calculated and stored as a double-word. The cube volumes must calculated and stored into the double-word sized array named ***cubeVolumes***.

All data must be treated as ***unsigned*** values (i.e., must use of MUL and DIV, not IMUL or IDIV).

Note, for an odd number of items, the middle value is defined as the middle value. For an even number of values, it is the integer average of the two middle values. The middle value is just the middle value of the array and as such is not statistically meaningful.

Do ***not*** change the sizes/types of the provided data sets. You may declare additional variables as needed.

Hint:

Pay close attention to the data types. The *sides[]* array and *cubeAreas[]* arrays are word sized, and the *cubeVolumes[]* array is double-word sized.

<u>Submission:</u>

- All source files must assemble and execute on Ubuntu with **`yasm`**.
- Submit source files
    - Submit a copy of the program source file via the on-line submission
- Once you submit, the system will score the project and provide feedback.
    - If you do not get full score, you can (and should) correct and resubmit.
    - You can re-submit an unlimited number of times before the due date/time.
- Late submissions will be accepted for a period of 24 hours after the due date/time for any given lab. Late submissions will be subject to a ~2% reduction in points per an hour late. If you submit 1 minute - 1 hour late -2%, 1-2 hours late -4%, … , 23-24 hours late -50%. This means after 24 hours late submissions will receive an automatic 0.

<u>Program Header Block</u>

All source files must include your name, section number, assignment, NSHE number, and program description. The required format is as follows:

```
;  Name: <your name>
;  NSHE ID: <your id>
;  Section: <section>
;  Assignment: <assignment number>
;  Description: <short description of program goes here>
```

Failure to include your name in this format will result in a loss of up to 5%.

<u>Scoring Rubric</u>

Scoring will include functionality, code quality, and documentation. Below is a summary of the scoring rubric for this assignment.

Criteria	**Weight**	**Summary**
Assemble	-	Failure to assemble will result in a score of 0.
Program Header	5%	Must include header block in the required format (see above).
General Comments	10%	Must include an appropriate level of program documentation.
Program Functionality (and on-time)	85%	Program must meet the functional requirements as outlined in the assignment. Must be submitted on time for full score.

Assignment #5 Provided Data Sets:

Use the following are the provided data declarations for assignment #5.

Note 1, a copy of the data set is provided on the class web site.
Note 2, the assembler **is** case sensitive.

```
; ----------------------------------------------------------------
;  Provided Data Set

sides       dw     10,  14,  13,  37,  54
            dw     14,  29,  64,  67,  34
            dw     31,  13,  20,  61,  36
            dw     14,  53,  44,  19,  42
            dw     44,  52,  31,  42,  56
            dw     15,  24,  36,  75,  46
            dw     27,  41,  53,  62,  10
            dw     33,   4,  73,  31,  15
            dw      5,  11,  22,  33,  70
            dw     15,  23,  15,  63,  26
            dw     16,  13,  64,  53,  65
            dw     26,  12,  57,  67,  34
            dw     24,  33,  10,  61,  15
            dw     38,  73,  29,  17,  93
            dw     64,  73,  74,  23,  56
            dw      9,   8,   4,  10,  15
            dw     13,  23,  53,  67,  35
            dw     14,  34,  13,  71,  81
            dw     17,  14,  17,  25,  53
            dw     23,  73,  15,   6,  13

length      dd    100

caMin       dw    0
caMid       dw    0
caMax       dw    0
caSum       dd    0
caAve       dw    0

cvMin       dd    0
cvMid       dd    0
cvMax       dd    0
cvSum       dd    0
cvAve       dd    0

; ----------------------------------------------------------------
;  Uninitialized data

section     .bss

cubeAreas         resw  100
cubevolumes resd  100
```

Note, the ".bss" section is for uninitialized data. The "resd" is used to reserve double-words and the "resw" is used to reserve words.