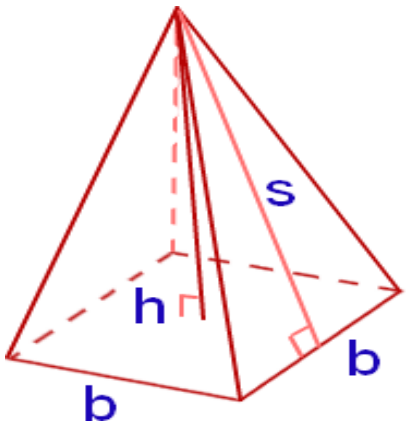
CS 218 – Assignment #5

Purpose: Learn to use arithmetic instructions, control instructions, compare instructions, and conditional jump instructions.
Points: 80

Assignment:

Write a simple assembly language program to calculate the some geometric information for each square pyramid[1] in a series of square pyramids. The program should find the lateral surface area (excluding base), total surface area, and volume for each square pyramid. Once the lateral surface areas, total surface areas, and volumes are computed, the program should find the minimum, maximum, estimated median value, sum, and average for the lateral surface areas, total surface areas, and volumes arrays.



All data must be treated as ***unsigned*** values (i.e., use of MUL and DIV, not IMUL or IDIV). The JA/JB/JAE/JBE must be used (as they are for unsigned data).

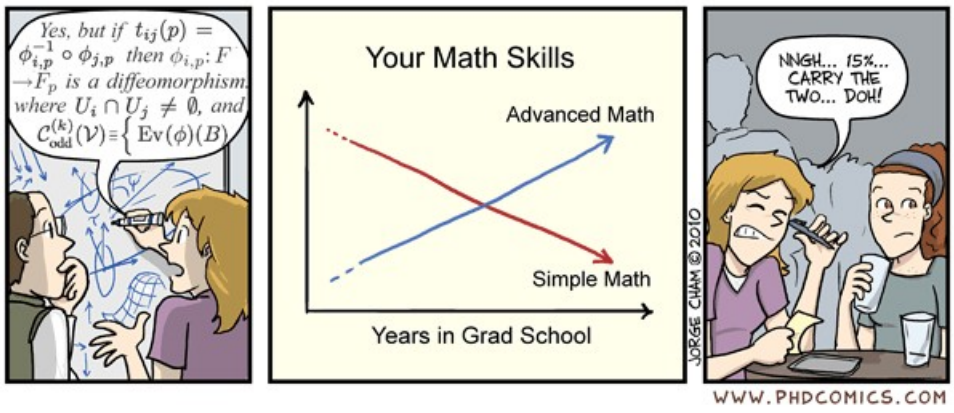
$$lateralAreas[n] = 2 \times bases[n] \times slants[n]$$

$$totalAreas[n] = bases[n] \times \left(2 \times slants[n] + bases[n] \right)$$

$$volumes[n] = \frac{bases[n]^2 \times heights[n]}{3}$$

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

Since the list is not sorted, we will estimate the median value. For project, the list length is odd, and the estimated median will be computed by summing the first, last, and the middle value and dividing by 3.



Note, no template is provided. Create the program source file based on the previous assignments.

Hints:

Pay close attention to the data types. The *bases[]* array is byte sized, the slants[] array is word sized, and the *heights[]* array is double-word sized.

Consider completing the lateral areas calculations before attempting the other calculations.

1 For more information, refer to: https://en.wikipedia.org/wiki/Square_pyramid

Submission:

- 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 assignment. 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.

Program Header Block

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: <4-digit-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 10%.

Scoring Rubric

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 Set:

Use the following provided data declarations for assignment #5.
Note, a copy of the data set is provided on the class web site.

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

bases         db      148,  194,  162,  163,  118
              db      161,  145,  152,  129,  165
              db      112,  100,  185,  163,  125
              db      176,  147,  155,  110,  113
              db      108,  145,  161,  164,  165
              db      177,  120,  156,  147,  161
              db      152,  119,  165,  161,  131
              db      165,  114,  123,  115,  114
              db      101,  171,  111

slants        dw      233,  214,  223,  211,  234
              dw      212,  200,  285,  263,  205
              dw      264,  213,  224,  213,  265
              dw      244,  212,  213,  212,  223
              dw      265,  264,  273,  216,  234
              dw      253,  213,  243,  213,  235
              dw      244,  169,  234,  233,  232
              dw      234,  223,  215,  214,  201
              dw      222,  242,  233

heights       dd      245,  234,  223,  223,  223
              dd      253,  253,  243,  253,  235
              dd      234,  234,  256,  264,  242
              dd      253,  253,  284,  242,  234
              dd      245,  234,  223,  223,  223
              dd      234,  234,  256,  264,  242
              dd      253,  253,  284,  242,  234
              dd      256,  264,  242,  234,  201
              dd      201,  223,  272

length        dd     43

laMin         dd     0
laEstMed      dd     0
laMax         dd     0
laSum         dd     0
laAve         dd     0

taMin         dd     0
taEstMed      dd     0
taMax         dd     0
taSum         dd     0
taAve         dd     0

vMin          dd     0
vEstMed       dd     0
vMax          dd     0
vSum          dd     0
vAve          dd     0

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

section       .bss

lateralAreas         resd   43
totalAreas           resd   43
volumes              resd   43
```

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