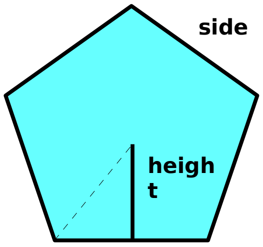
CS 218 – MIPS Assignment #3

Purpose:	Become familiar with the MIPS stack and procedure calling convention.
Points:	100

Assignment

Write a MIPS assembly language program to calculate the area of each regular pentagon[1] in a series of regular pentagons. Use the provided MIPS main program and develop the following functions:

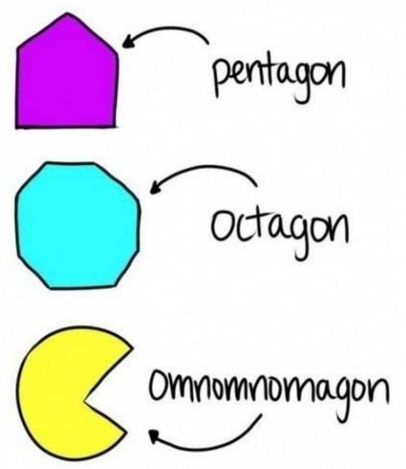


- Write a MIPS void function, ***calcAreas()***, to calculate the area for each pentagon in a series of pentagons. The formula for the area is as follows:

$$\boldsymbol{areas[i] = \frac{5 \times sides[i] \times heights[i]}{2}}$$

- Write a MIPS void function, ***areasStats()***, that will find the minimum, maximum, median, integer average, and float average of the diagonals array. The function must call the ***iSum()***, ***iAverage()***, and ***fAverage()*** functions. Additionally, the routine must call the ***insertionSort()*** function before the median is calculated.

- Write a MIPS void function, ***insertionSort()***, to sort the areas array in descending order (large to small). To sort the numbers, use the following Insertion Sort[2] algorithm:



```
insertionSort(array arr) {
   for i = 1 to length-1 do  {
      value = arr[i];
      j = i - 1;
      while((j ≥ 0) and (arr[j] > value)){
          arr[j+1] = arr[j];
          j = j - 1;
      }
      arr[j+1] = value;
   }
}
```

You must use the above Insertion Sort algorithm (i.e., do not use a different sort). *Note*, the algorithm assumes array index's start at 0. As necessary, you can define additional variables. ***Submissions not based on this algorithm will not be scored***.

- Write a value returning MIPS function, ***iSum()***, to find the sum of an array.

[1] For more information, refer to: https://en.wikipedia.org/wiki/Pentagon

[2] For more information, refer to: https://en.wikipedia.org/wiki/Insertion_sort

- Write a MIPS value returning function, ***iAverage()***, to find the integer average of an array. The function must call the ***iSum()*** function.

- Write a MIPS value returning function, ***fAverage()***, to find the floating point average of an array. The function must call the ***iSum()*** function and perform the required type conversions.

- Write a MIPS void function, ***printResults()***, to print the areas array (eight per line) and the statistical information (minimum, maximum, median, integer average, and float average) in the format shown in the example.

<u>Example Output</u>

The program must display the results to the console window. The output should look something like the following (with all of the correct answers displayed for all data sets):

```
MIPS Assignment #3
Pentagon Areas Program


********************************************************************
Pentagon Data Set #1
Length: 30

Areas - Values:
     15360     14822     14760     14687     14490     14190     13867     13837
     13335     13335     13310     12812     12810     12400     12000     11210
     10822     10735     10637     10440     10115     9887     9817     9435
     9360     9360     8960     8912     8550     8250

Areas - Stats:
   sum  = 352505
   min  = 8250
   max  = 15360
   med  = 11605
   int ave = 11750
   flt ave = 11750.16699219


********************************************************************
Pentagon Data Set #2
Length: 150

      . . .   output truncated . . .
```

Note, the numbers do not need to be aligned. You will need to print spaces between each number (or they will all run together). Check the output in CodeGrade to verify that your spacing matches the expected output

Submission

- All source files must assemble and execute with QtSpim/SPIM MIPS simulator.
- Submit source file
  - 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 (at a maximum rate of 5 submissions per hour).
- 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: <section>
#  Assignment: <assignment number>
#  Description: <short description of program goes here>
```

Failure to include your name in this format will result in a reduction of points.

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	3%	Must include header block in the required format (see above).
General Comments	7%	Must include an appropriate level of program documentation.
Program Functionality (and on-time)	90%	Program must meet the functional requirements as outlined in the assignment. Must be submitted on time for full score.