Problem. Write a program that uses structures and pointers. You will have to write two functions: **get_stats**, and **get_median**.

(1) First in lab7.c, you need to declare a structure type **driver_t**.
I named my structure **driver_t** and its 4 parts are:
a character array **name** that is 21 in length, (comes from the data file)
a double array of **tries** that has a length of TRIES, (comes from the data file)
a double named **best_time**, (value computed by program)
a double named **deviation**, (value computed by program).

Name	**Try 0**	**Try 1**	**Try 2**	Best Time	Deviation

Array

(2) Next in lab7.c, you need to declare a structure type:_ **stats_t**.
I named my structure **stats_t** and its 4 parts are:
four variables, all type double, named **best_average**, **fast_time**, **slow_time**, and **median**.

(3) Write the function **get_stats**. This function will figure the driver's best time, the track slow time and fast time, the average of the driver's best times, and the driver's deviation from the fast time. The prototype is:

```
void get_stats(driver_t driver_list[NRACERS],      /* in & out */
               stats_t *race_stats );               /* in & out */
```

(4) Write the function **get_median**. It will find the mid best time from the sorted list of best times. Examples of computing median are on the top of page 4. The prototype for **get_median** is:

```
void get_median(driver_t driver_list[NRACERS],  stats_t *race_stats );
```

(5) You will be provided a test driver program that needs ALMOST NO changing, only ADDing.
You will only need to **add** the two structures and the two functions as above.
You will need to shift the comment marks (//) on the four #define statements: two for the data file, two for the NRACERS

Input/Output Description:
The program input is a set of driver's names and their three tries on the race track in one file. The race times are type double. Each record/line of the file has a student name and three times.

The first line from the sample data file is:

```
Jay Johnson          4.100     5.300     6.700
```

The output is printed to **lab7.out** as shown in the sample output.

Algorithm Development - Pseudo code:

```
/*-----------------------------------------------------------*/
main
  /* This function already exists. */
  out_file = open_out_file ();
  get_data(IN_FILENAME, driver_list);
  get_stats(driver_list, &race_stats);
  do_sort(driver_list);
  get_median(driver_list, &race_stats);
  print_all(out_file, driver_list, &race_stats);

/*-----------------------------------------------------------*/
FILE * open_out_file(void)
   /* This function already exists. */
   /* Opens the output file */

/*-----------------------------------------------------------*/
void get_data (char *filename,              /* input  */
                  driver_t driver_list[NRACERS] );  /* output */

   /* This function already exists. */
   /*It opens the data file and reads it into the appropriate places. */

/*-----------------------------------------------------------*/
void print_all(FILE * out_file,
                driver_t driver_list[NRACERS] ,
                stats_t *race_stats )

   /* This function already exists. */

/*-----------------------------------------------------------*/
void do_sort(student_t student_list[NSTUDENTS])

   /* This function already exists. */

/*-----------------------------------------------------------*/
```

➔ more on next page

```
/*----------------------------------------------------------*/
/*    THIS IS A SUB-FUNCTION THAT YOU HAVE TO WRITE  */
void get_stats( driver_t driver_list[NRACERS],    /* in & out */
                   stats_t *race_stats )                /* in & out */

  Zero out the best_average  (HINT: use the -> notation)
  Set the slow_time to the first driver's first try.
  Set the fast_time to the first driver's first try.

  loop from d=zero to < NRACERS increment by one
 {
         zero out the driver_list[d].deviation
         set the driver's best time to the driver's first time
         loop from t=zero to t< TRIES increment by one
         {
             figure the driver's best time
              find the fastest and slowest track time
         }
         add the driver's best time into the running total of best times
 }
  compute the average of the best times

  loop from d=zero to < NRACERS increment by one
  {
         figure the driver's deviation from the fast_time(the fastest all-over time)
         (deviation is fast time minus driver's best time)
  }
  return

/*----------------------------------------------------------*/
/*  THIS IS A SUB-FUNCTION THAT YOU HAVE TO WRITE */
void get_median(driver_t driver_list[NRACERS],
                     stats_t *race_stats )

   zero out the median.
   calculate the mid point (divide NRACERS by two)
   if the number of racers is odd then
      set the median to the mid average
   else
      set the median to the average of the two numbers(averages) on
             each side of the median. [mid] & [mid-1]. NO integer division.

/*----------------------------------------------------------*/
```

➔ **Examples of Median on next page.**

NOTES on the median:
The median is the value in the middle of a group of values, assuming that the values are sorted. If there is an odd number of values, the median is the value in the middle. If there is an even number of values, the median is the average of the values in the two middle positions.
EXAMPLES:

(1) The median of values {1, 6, **18**, 39, 86} is the middle value, or 18.
(2) The median of values {1, 6, **18, 39**, 86, 91} is the average of the two middle values, or (18 + 39)/2 or 28.5.

Sample Data:
This is the sample data example. It does not match the lab7.dat file in length **or** in value!

SAMPLE DATA:

```
Jay Johnson                  4.100        5.300        6.700
Missy Monroe                 1.000        2.000        3.500
Ned Niner                    3.800        7.000        5.500
Lenny Loop                   2.200        3.400        4.600
```

Sample Output :

```
Your Name.  Lab 7 output.

Track Results

Drivers                    Try 1       Try 2       Try 3      Best Time   Deviation
--------------------     ---------   ---------   ---------   ----------   ---------
Missy Monroe                1.000       2.000       3.500        1.000       0.000
Lenny Loop                  2.200       3.400       4.600        2.200      -1.200
Ned Niner                   3.800       7.000       5.500        3.800      -2.800
Jay Johnson                 4.100       5.300       6.700        4.100      -3.100


 The average of best times =    2.775

 The track fast time       =    1.000

 The track slow time       =    7.000

 The median of best times  =    3.000
```

Using the Sample Data:
To use the sample date, you need to make changes:
These lines are currently set to access the sample file. *Move the comment symbols (//) from one line to another to shift the use of files.*

```
//#define IN_FILENAME "lab7.dat"
#define IN_FILENAME "lab7sample.dat"
```

The final data file has a length of 10, the sample file has a length of 4. *Move the comment symbols (//) from one line to another to shift the use of files.*

```
//#define NRACERS 10
#define NRACERS 4
```

→ **more on next page**

__Files To Copy__:

Type: **cp -R /gaia/home/faculty/bielr/files_csc60/lab7 .**

Spaces needed: (1) After the **cp** ↑ *Don't miss the space & dot.*

(2) After the **-R**

(3) After the directory name at the end & before the dot.

After the files are in your account and you are still in **csc60**,
you need to type: **chmod 755 lab7**
This will give permissions to the directory.
Next move into lab7 directory by typing: **cd lab7**
After the files are in your account, you need to type: **chmod 644 lab7***
This will give permissions to the files.
Your new lab7 directory should now contain: lab7.c, lab7.dat, lab7sample.dat

__PREPARE YOUR FILE FOR GRADING:__

Make sure your program has been:

- corrected to use **lab7.dat**
- corrected to use the proper value for **NRACERS**
- has been re-complied.

When all is well and correct,
Type: **script StudentName_lab7.txt** [Script will keep a log of your session.]
Type: **a.out** or **lab7** to run the program to show the output of the program
(or __whatever name__ you used for the executable)
Type: **cat lab7.out** to see the output of your program
Type: **exit** to leave the script session

__Turn in your completed session:__

Go to Canvas and turn in (no zip files):

1. lab7.c
2. your script session (StudentName_lab7.txt)